

## Image Captioning System Using Artificial Intelligence

Author

Saba Syed

### Abstract

*People who are blind or visually impaired often find it difficult to fully engage with the world around them, as they cannot accurately perceive the visual information that sighted people take for granted. As a result, they often require some form of human assistance to navigate their environment and access information. In this project, we aim to use image captioning techniques to help solve this problem. By leveraging a large dataset and machine learning algorithms, we hope to improve the ability to convert captured and stored images into text and speech that can be easily understood by blind individuals. Our project is inspired by recent advancements in multimodal neural networks, which have been successfully used in image captioning systems. Specifically, we will use a combination of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to translate images into text. The CNN will be used for visual feature extraction, while the RNN will be trained on image-sentence ground truth pairings to generate captions. One of the main challenges we face is the language barrier. While there have been numerous studies on image captioning for single-target languages, we aim to develop a system that can generate captions in multiple languages. To achieve this, we will use the GoogleTrans library, which implements the Google Translate API. Our project will follow the AI essentials framework for designing AI products, as well as the Scrum methodology for managing the software development lifecycle. We will begin by collecting a dataset of 8,000 images from the Flickr 8k dataset, and use OCR Tesseract to extract text from any images that contain text. We will then use a pre-trained CNN as a feature extractor, and feed these features into an LSTM, which will generate captions. These captions will be translated into multiple languages using the GoogleTrans library, and finally converted to speech using the gTTS library. Overall, our project aims to improve the lives of blind and visually impaired individuals by providing them with a more accurate and comprehensive understanding of their surroundings. By leveraging machine learning and neural networks, we hope to develop a system that is capable of generating accurate and useful captions in multiple languages, thereby bridging the language barrier and making it easier for blind individuals to access information.*

**Keywords:** Image captioning system, artificial intelligence, machine learning, neural networks.

## Introduction

### 1.1. Brief

In Artificial Intelligence (AI), the contents of an image are generated automatically which involves computer vision and NLP (Natural Language Processing). The neural model which is regenerative, is created. It depends on computer vision and machine translation. This model is used to generate natural sentences which eventually describes the image. This model consists of Convolutional Neural Network (CNN) as well as Recurrent Neural Network (RNN). The CNN is used for feature extraction from image and RNN is used for sentence generation. The model is trained in such a way that if input image is given to model it generates captions which nearly describes the image. The accuracy of model and smoothness or command of language model learns from image descriptions is tested on different datasets. These experiments show that model is frequently giving accurate descriptions for an input image.

Image processing is a growing subject of study. Images come in a variety of file formats and represent a variety of things, including places, persons, scientific, astrological, and other topics. An image is made up of a series of pixels that are arranged in rows and columns. These photos are taken, processed, and saved for a variety of purposes. It is simple for ordinary individuals to identify and interpret generic images, but it is more challenging for the blind and physically disabled. Unfortunately, such needy people have no prior method or interface through which to communicate with the rest of the world. People who are blind or visually impaired are sometimes neglected by society, so there is always a need to assist them, Mrunmayee Patil et al[1].

Caption generation is a challenging artificial intelligence problem where a textual description must be generated for a given photograph. It requires both methods from computer vision to understand the content of the image and a language model from the field of natural language processing to turn the understanding of the image into words in the right order. Recently, deep learning methods have achieved state-of-the-art results on examples of this problem.

Deep learning methods have demonstrated state-of-the-art results on caption generation problems. What is most impressive about these methods is a single end-to-end model can be defined to predict a caption, given a photo, instead of requiring sophisticated data preparation or a pipeline of specifically designed models.

Recent computer vision research has produced excellent achievements in autonomously describing images using natural language. To create a multilingual captioning system, most of these systems are generated in a single language, requiring using numerous language-specific models. We suggest a fairly basic way for creating a single uniform model across multiple platforms artificial intelligence used to govern the language. It is a cost-effective and efficient approach for visually impaired people to hear text graphics in their own languages. Text-to-speech conversion is a technique that scans and reads any language letters and numbers in an image, then translates it into any desired language and outputs the translated text as audio.

## **1.2. Project Background**

People who are blind or visually impaired find it difficult to engage with the world, and because they cannot precisely sense things, they require some form of human intervention. In this project, image captioning techniques are used to solve this problem as a large dataset will be trained and tested. The suggested system improves the ability to convert captured and stored images into text and speech. For blind people who are unable to perceive visuals, the introduction of machine learning technologies such as image description is a blessing. Image descriptions can be read out using an AI-powered image caption generator, allowing them to obtain a better feel of their surroundings.

## **1.3. Literature Review**

Multimodal neural networks, inspired by sequence-to-sequence modeling in machine translation, are used in the most recent image captioning systems, I. Sutskever et al [1].

Images are loaded into a Convolutional Neural Network (CNN) for visual feature extraction, and then translated to word sequences using a Recurrent Neural Network (RNN) trained on image sentence ground truth pairings, j. Johnson et al [3].

Although numerous alternatives of multi-language captioning data have been examined, most of this work on image captioning has focused on a single target language (English).

Elliott et al. [4] and Miyazaki et al.[5], for example, examine the difficulty of image captioning for one language when another language's caption is available.

#### **1.4. Analysis from Literature Review**

The research papers above explain how the images are translated into text using CNN and RNN. Along with the problem of language barrier, e.g., the captions are created in a single language that is English.

#### **1.5. Methodology and Software Lifecycle for This Project**

For this system, we will use a combination of scrum and the AI essentials framework.

IBM created the AI essentials framework, and SCRUM is readily accessible at [scrum.org](https://www.scrum.org).

AI essentials framework is used for designing AI products.

ML are complex technologies that take time to implement and fully leverage and Scrum is a more disciplined, prescribed approach to deal with complexity and uncertainty.

##### **Lifecycle**

The lifecycle of our project is as following:

- **Data Collection**

There are many open-source datasets available for problems like these, like Flickr 8k, Flickr 30k, MS COCO, etc. We'll be using Flickr 8k [6] dataset, which contains 8k images.

- **Text Extraction**

With the help of OCR Tesseract, we will extract the text from the images containing text and store it in a temporary variable. This text will be later used for text to speech module.

OCR systems transform images of text, that contain machine printed or handwritten text from its image representation into machine-readable text. OCR as a process generally consists of several sub-processes to perform as accurately as possible.

- **Feature Extraction (encoding)**

A CNN (Convolutional Neural Network) is pre-processed for image (non-textual) classification task. This pre-trained CNN will be used as a feature extractor and these features will then be used as an input for LSTM which works as a decoder.

- **Caption Generation (decoding)**

LSTM (Long Short-Term Memory) are used as the building blocks for the layers of an RNN. LSTM will then assign data weights that will help to either give enough importance

to impact the output, forget the information or let new information in. The caption will then finally be generated with the help of LSTM.

- **Text Translation:**

This textual description will then be translated into any languages requested by the user. The translation in multiple languages is possible with the help of GoogleTrans library. It is a free python library that implements Google Translate API.

- **Text-to-speech**

By using gTTS (Google text-to-speech) library, the translated caption or the text from a textual image, will then be converted into an audio format. This will be the final output of our system.

## **Problem Statement**

### **2.1. Purpose**

The purpose of our system is to help blind community to understand images as it will be generating captions according to activity that is happening in the image, and then it will convert those captions into speech for better understanding of the blind people. It will also extract the text, if any, from the images and convert it into speech.

### **2.2. Product Functions**

The most important function of our system is that it will generate natural language descriptions according to the content observed in an image. It is an important part of scene understanding, which combines the knowledge of computer vision and natural language processing. As we are targeting the blind community, our system will translate the image into textual description and then into speech by using Artificial intelligence techniques like CNN and LSTM.

### 2.3. Proposed Architecture

The architecture that we are using is CNN-LSTM architecture. This architecture involves using Convolutional Neural Network (CNN) layers for feature extraction on input data combined with LSTMs to support sequence prediction.

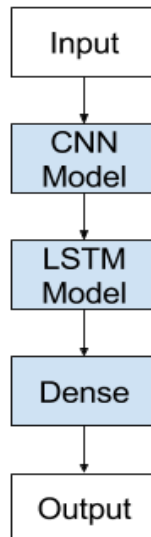


Figure 2.1 Generic architecture of a CNN LSTM Model

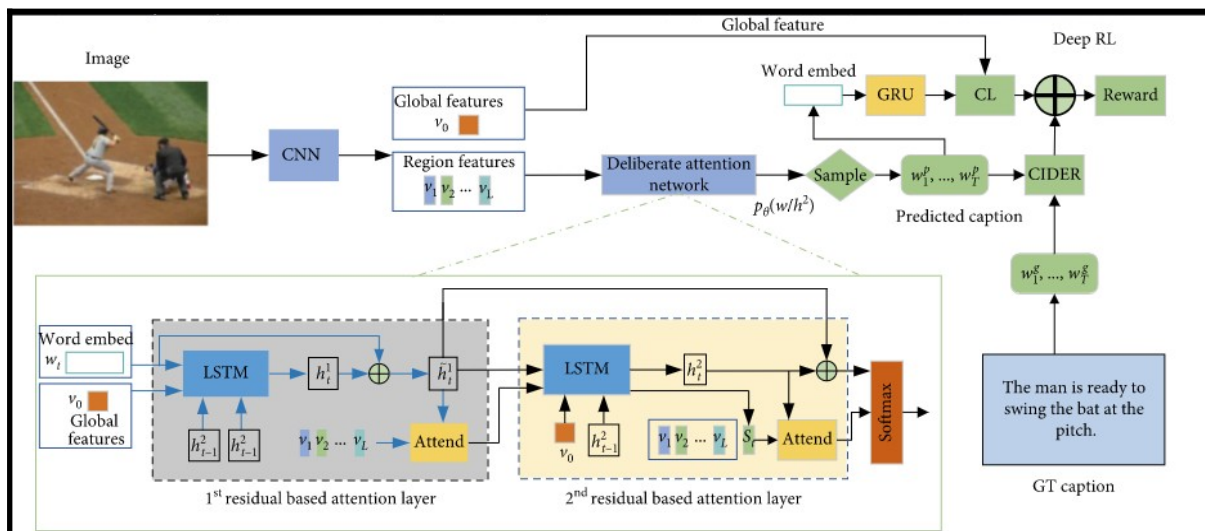


Figure 2.2 In-depth CNN-LSTM Architecture

### 2.4. Project Deliverables

Project deliverables are the milestones we cover in the whole project. These are:

- For Feature Extraction (encoding) a CNN model is used which is pre-processed for image (non-textual) classification tasks. These features will then be used as an input for LSTM which works as a decoder.
- For Caption Generation (decoding) LSTM are used as the building blocks for the layers of an RNN. LSTM will then assign data weights that will help to either give enough importance to impact the output, forget the information or let new information in. The caption will then finally be generated with the help of LSTM.
- Then comes text translation. Textual description will then be translated into any languages requested by the user. The translation in multiple languages is possible with the help of GoogleTrans library. It is a free python library that implements Google Translate API.
- Text-to-speech by using gTTS (Google text-to-speech) library, the translated caption or the text from a textual image, will then be converted into an audio format. This will be the final output of our system. The architecture that we are using is CNN-LSTM architecture. This architecture involves using Convolutional.

## **2.5. Operating Environment**

The libraries and tools that play a major role in developing this system are:

- Keras is an open-source ML library that's written in Python. It builds and trains neural networks. It is user friendly, modular and highly flexible and extendable.
- An OCR tool for python to organize and read the text embedded in images.
- GTTS is a Python library and CLI tool to interface with Google Translate Text to speech API.

The database we will use is MySql and the operating system needed for this system is Windows10.

The minimum requirement of hardware needed for this system is:

- GPU: GTX 1650 or 1660 TI
- CPU: i5 9th or 10th Gen.
- RAM: 16GB

## **2.6. Assumptions & Dependencies**

The architecture we are using for this system is CNN LSTM which is way more efficient than any other. The quality of audio is not compromised either, the user will not have any issue regarding the text to speech module. There is a variety of languages to choose so users from all over the world will be able to make use of this software.

The dependencies of this system are:

- Our system is reliant on technology in order to develop the project, if the AI technology advances, we will need to restructure the current system to accommodate the new technology.
- The system is team-dependent since it requires collaboration to create the solution in AI and Python libraries.

# **Software Requirement Analysis**

## **3.1. Use Cases**

Use cases are a widely used and highly regarded format for capturing requirements. Before writing functional requirement use cases can help you to understand the requirements in the way user expect.



Use Case Diagram:

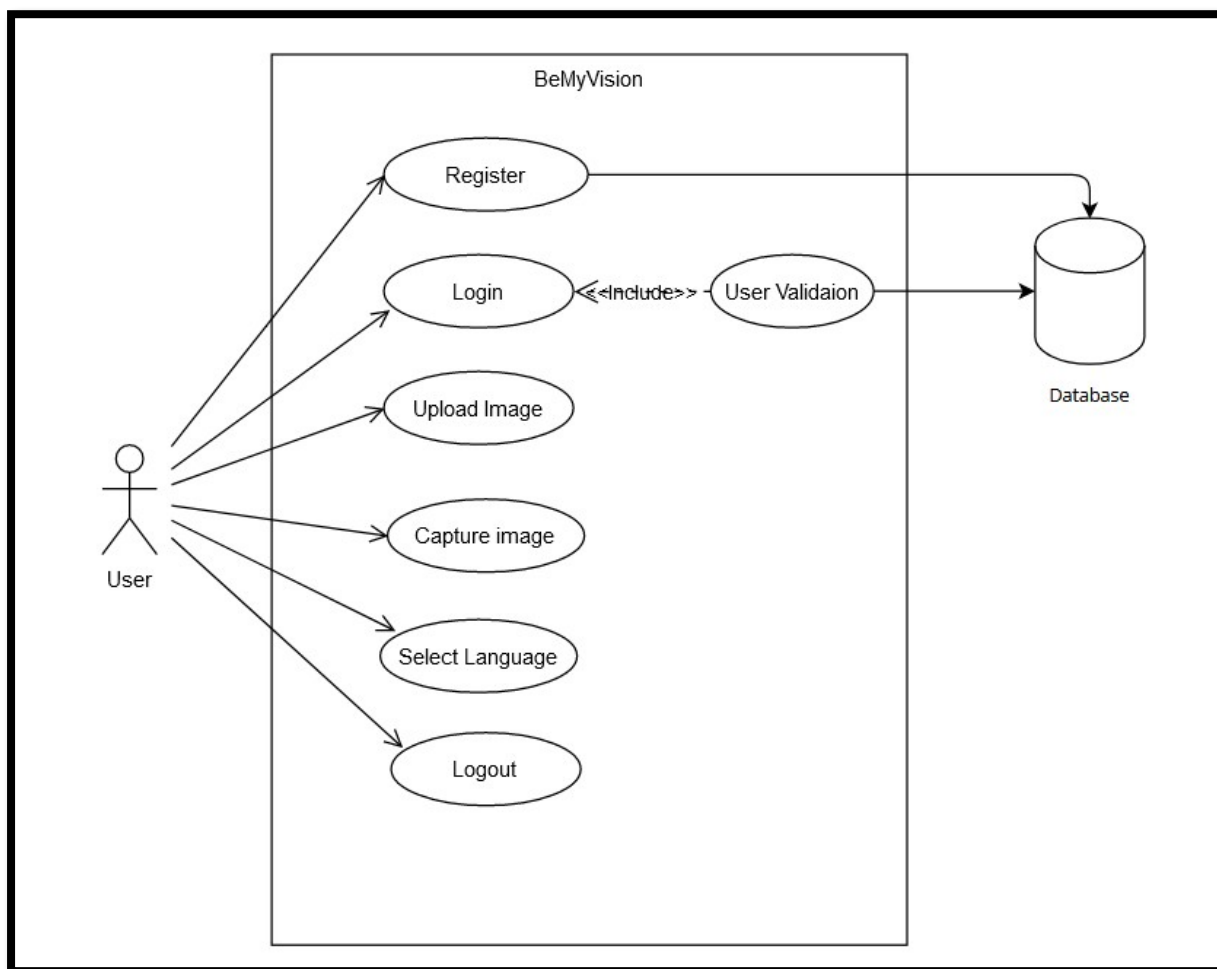


Figure 3.1 Use Case Diagram

**Use Case Description:**

*Table 3.1 Registration*

<b>Use Case:</b>	01		
<b>Use Case Name:</b>	<b>Register</b>		
<b>Created By:</b>	Saba Syed	<b>Last Updated By:</b>	25-DEC-2021
<b>Date Created:</b>	25-DEC-2021	<b>Last Revision Date:</b>	
<b>Actors:</b>	User		
<b>Description:</b>	User can register		
<b>Trigger:</b>	Register button		
<b>Preconditions:</b>	<b>User must open the application</b>		
<b>Post conditions:</b>	Registered successfully		
<b>Normal Flow:</b>	Actor	System	
	User registers himself	Register on application	
<b>Alternative Flows:</b>	User closes the window		
<b>Exceptions:</b>	If the system crashes after turning on it will start from here		

Table 3.2 Login

<b>Use Case ID:</b>	02		
<b>Use Case Name:</b>	Login		
<b>Created By:</b>	Saba Syed	<b>Last Updated By:</b>	25-DEC-2021
<b>Date Created:</b>	25-DEC-2021	<b>Last Revision Date:</b>	
<b>Actors:</b>	User		
<b>Description:</b>	User can Login		
<b>Trigger:</b>	Login button		
<b>Preconditions:</b>	User must be registered		
<b>Post conditions:</b>	Login successfully		
<b>Normal Flow:</b>	<b>Actor</b>	<b>System</b>	
	User must login	Login the user	
<b>Alternative Flows:</b>	User closes the window		
<b>Exceptions:</b>	If the system crashes after turning on it will start from here		

Table 3.3 Browse Image

<b>Use Case ID:</b>	03		
<b>Use Case Name:</b>	Browse Image		
<b>Created By:</b>	Saba Syed	<b>Last Updated By:</b>	25-DEC-2021
<b>Date Created:</b>	25-DEC-2021	<b>Last Revision Date:</b>	
<b>Actors:</b>	User		
<b>Description:</b>	User can Browse image		
<b>Trigger:</b>	Browse image button		
<b>Preconditions:</b>	User must be logged in		
<b>Post conditions:</b>	Browsed image successfully		
<b>Normal Flow:</b>	<b>Actor</b>	<b>System</b>	
	User must browse image	Search image in the system	
<b>Alternative Flows:</b>	User closes the window		
<b>Exceptions:</b>	If the user closes file explorer without selecting any image the system will not crash and generate an error#404 dialogue box.		

*Table 3.4 Select Language*

<b>Use Case ID:</b>	04		
<b>Use Case Name:</b>	Select Language		
<b>Created By:</b>	Saba Syed	<b>Last Updated By:</b>	26-DEC-2021
<b>Date Created:</b>	26-DEC-2021	<b>Last Revision Date:</b>	
<b>Actors:</b>	User		
<b>Description:</b>	User can select language		
<b>Trigger:</b>	Select language button		
<b>Preconditions:</b>	User must be logged in		
<b>Post conditions:</b>	Language selected successfully		
<b>Normal Flow:</b>	<b>Actor</b>	<b>System</b>	
	User must select language	Select the language which user picked	
<b>Alternative Flows:</b>	User closes the window		
<b>Exceptions:</b>	If the system crashes after turned on it will start from here		

Table 3 5 Text-to-Speech

<b>Use Case ID:</b>	05		
<b>Use Case Name:</b>	Capture Image		
<b>Created By:</b>	Saba Syed	<b>Last Updated By:</b>	27-DEC-2021
<b>Date Created:</b>	27-DEC-2021	<b>Last Revision Date:</b>	
<b>Actors:</b>	User		
<b>Description:</b>	User can Capture image directly from web camera		
<b>Trigger:</b>	Capture Image button		
<b>Preconditions:</b>	User must be logged in		
<b>Post conditions:</b>	Image captured and displayed successfully		
<b>Normal Flow:</b>	Actor	System	
	User must capture image button	Capture image using web camera and display the image	
<b>Alternative Flows:</b>	User closes camera		
<b>Exceptions:</b>	If the user closes camera without capturing image, main screen will be displayed		

Table 3.6 Login

<b>Use Case ID:</b>	06		
<b>Use Case Name:</b>	Logout		
<b>Created By:</b>	Saba Syed	<b>Last Updated By:</b>	25-DEC-2021
<b>Date Created:</b>	25-DEC-2021	<b>Last Revision Date:</b>	
<b>Actors:</b>	User		
<b>Description:</b>	User can Logout		
<b>Trigger:</b>	Logout button		
<b>Preconditions:</b>	User must be logged in		
<b>Post conditions:</b>	Logout successfully		
<b>Normal Flow:</b>	<b>Actor</b>	<b>System</b>	
	User must logout	Logout the user and display Login Screen	
<b>Alternative Flows:</b>	User closes the window		
<b>Exceptions:</b>	If the system crashes, the application will start from the login screen.		

### 3.2. Design Constraints

Design constraints help the users to know about the app, how it works and how much it is accurate and efficient.

- **Accuracy**  
Using best ML techniques to ensure best accuracy
- **Performance**  
Works on both Windows and Linux as desktop application
- **Design Methodology**

We used agile methodology for our project. The goal is continuous improvements and customer satisfaction

- **User View**

At the start of the application, Login screen is displayed where the user can sign in if he already has an account otherwise on selection of the Signup button, registration screen will be displayed where the user can create a new account. On successful login, the main screen will be opened where the user can upload an image and captions will be displayed.

### 3.3. Functional Requirements

Functional requirements are the detailed services that the software offers. The functional requirements for “BeMyVision” are:

- **Register:**

The user should register himself before login.

- **Login:**

The user should be logged in the system before browsing an image.

- **Browse Image:**

The user should browse an image from an image library or somewhere the images are stored.

- **Capture Image:**

The user can capture an image from the web camera.

- **Select Languages:**

After Browsing an image trained model then identifies the language which it will have to speak.

- **Select Text-To-Speech:**

Select Text-to-speech is for listening to the caption/text generated from the image.

### 3.4. Non-Functional Requirements

Non-functional requirements are constraints and restrictions on the design of the software that ensure the usability and effectiveness of the said software. The non-functional requirements for “BeMyVision” are:

- **Availability**



Internet availability is required.

- **Security**

User accounts are secured through passwords since each user will have his/her own account

- **Useability**

Efficient for carrying out all the tasks

- **Performance**

Performance is dependent on hardware

## Design and Architecture

This chapter will discuss the design and architecture of our system.

### 4.1. System Architecture

As system design varies from system to system, therefore user needs to have the architecture view of the whole system.

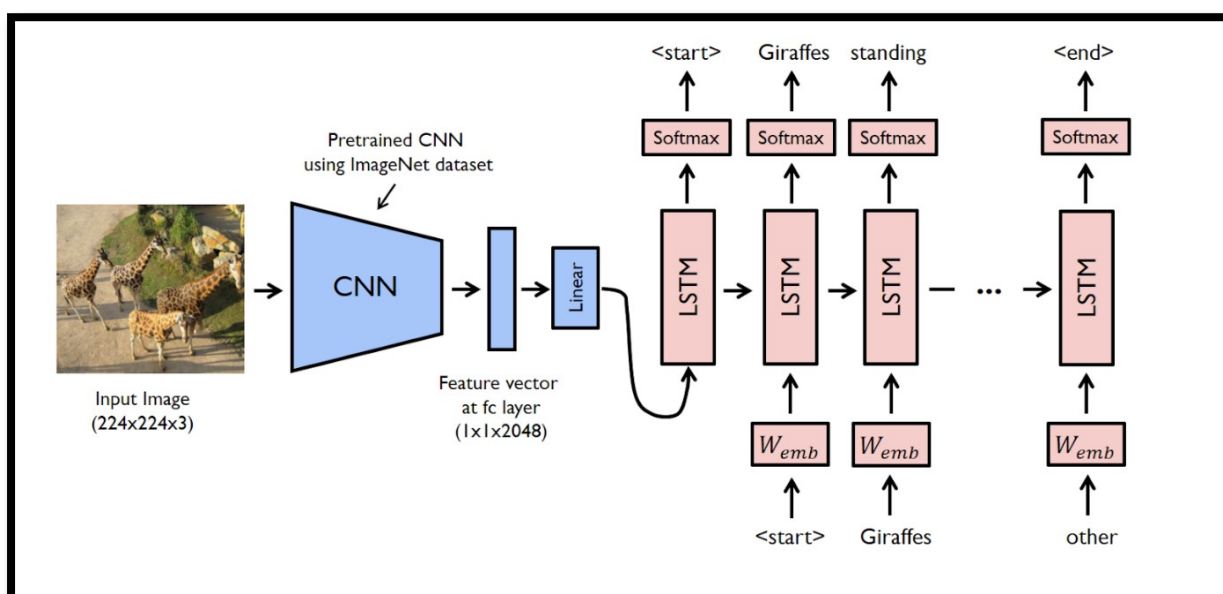


Figure 4.1 System Architecture

### 4.2. System Design

Systems design is the process of defining elements of a system like components, modules, architecture and their interfaces and data for a system based on the specified requirements.

The purpose of the System Design process is to provide sufficient detailed data and information about the system.

### UML Structural Diagrams

Following are the UML structural diagrams of our system:

#### 4.2.1.1 Component Diagram

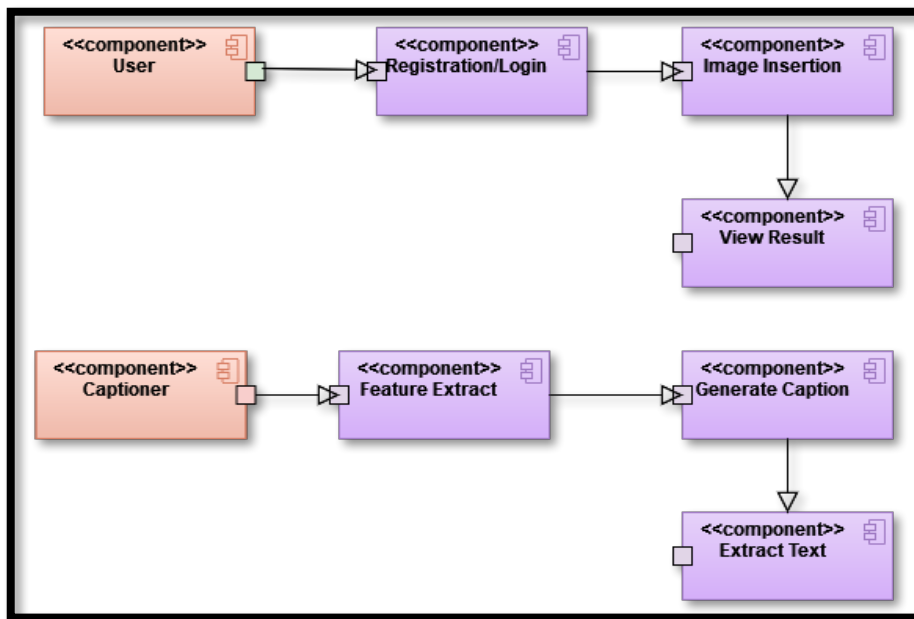


Figure 4.1 Component Diagram

#### 4.2.1.2. Package Diagram

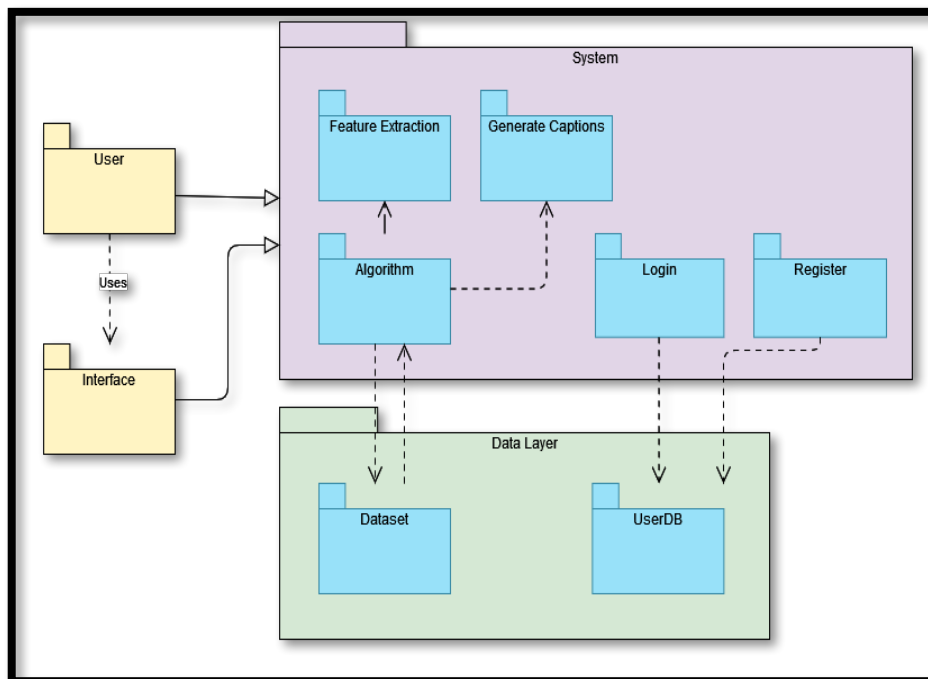


Figure 4.2 Package Diagram

## UML Behavioral Diagrams

Following are the behavioral diagrams of our system:

### 4.2.2.1 Activity Diagram

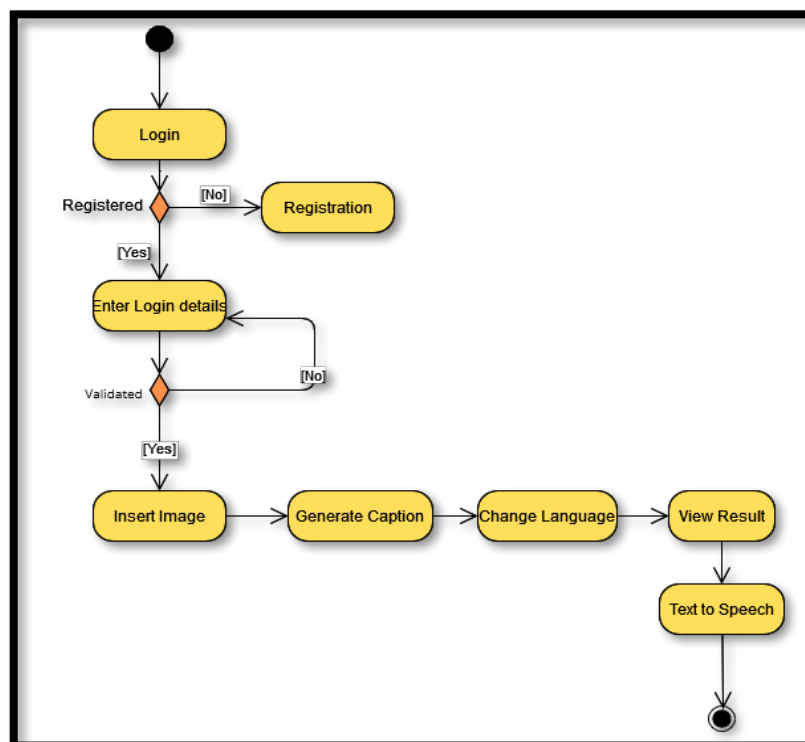
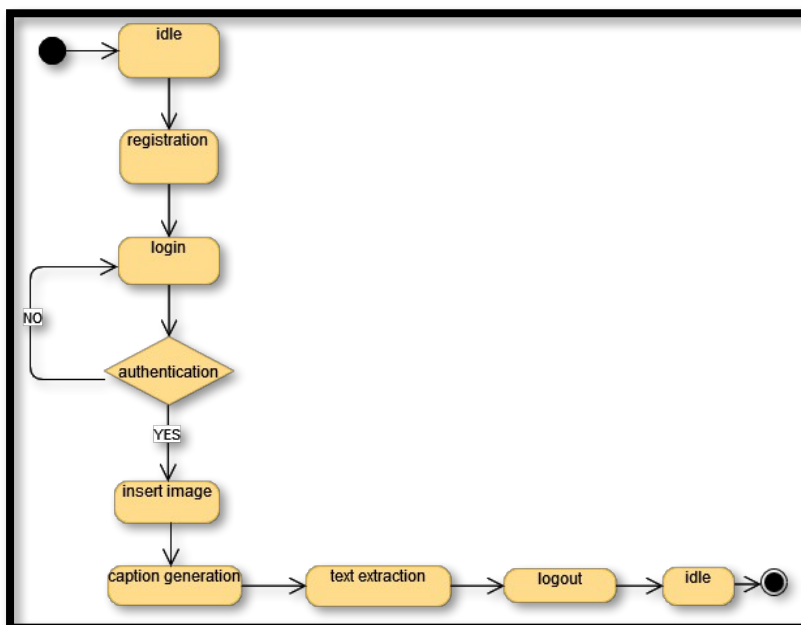


Figure 4.3 Activity Diagram

### 4.2.2.2 State Machine Diagram



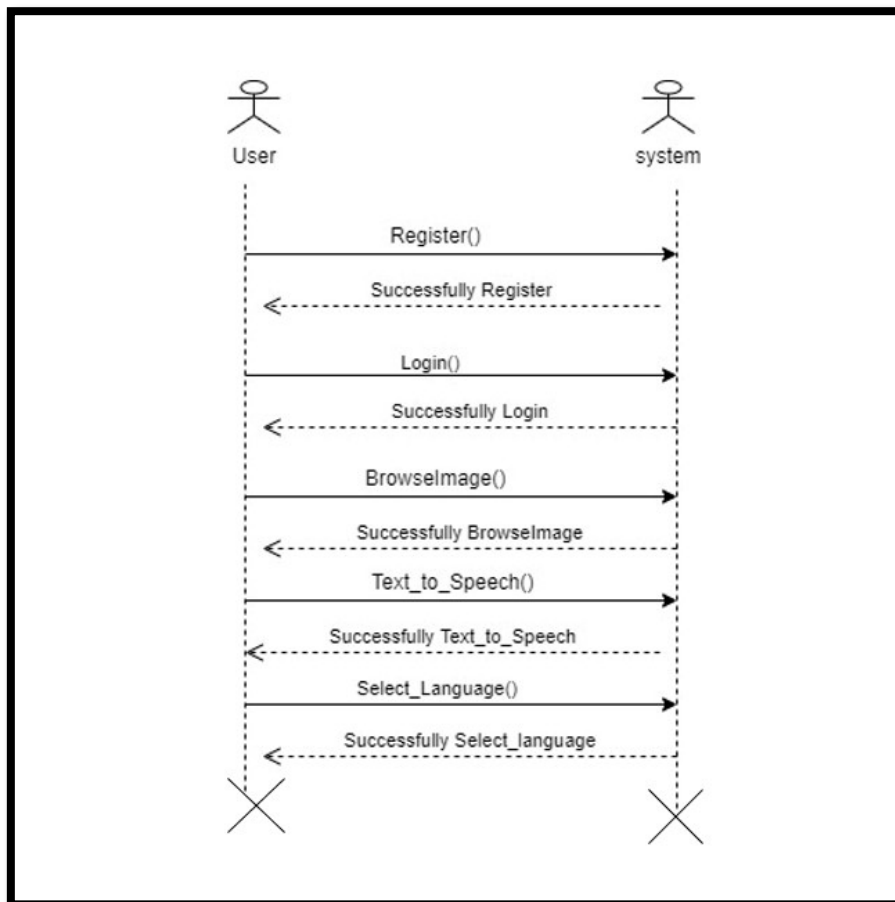
*Figure 4.4 State Machine Diagram*

## UML Interaction Diagrams

Following are the UML interaction diagrams of our system:

### 4.2.3.1 Sequence Diagram

A sequence diagram is a type of interaction diagram because it describes how and in what order a group of objects works together.



*Figure 4.5 Sequence diagram*

## Implementation

This chapter will discuss implementation details followed by user interface.

### 5.1. Component Diagram

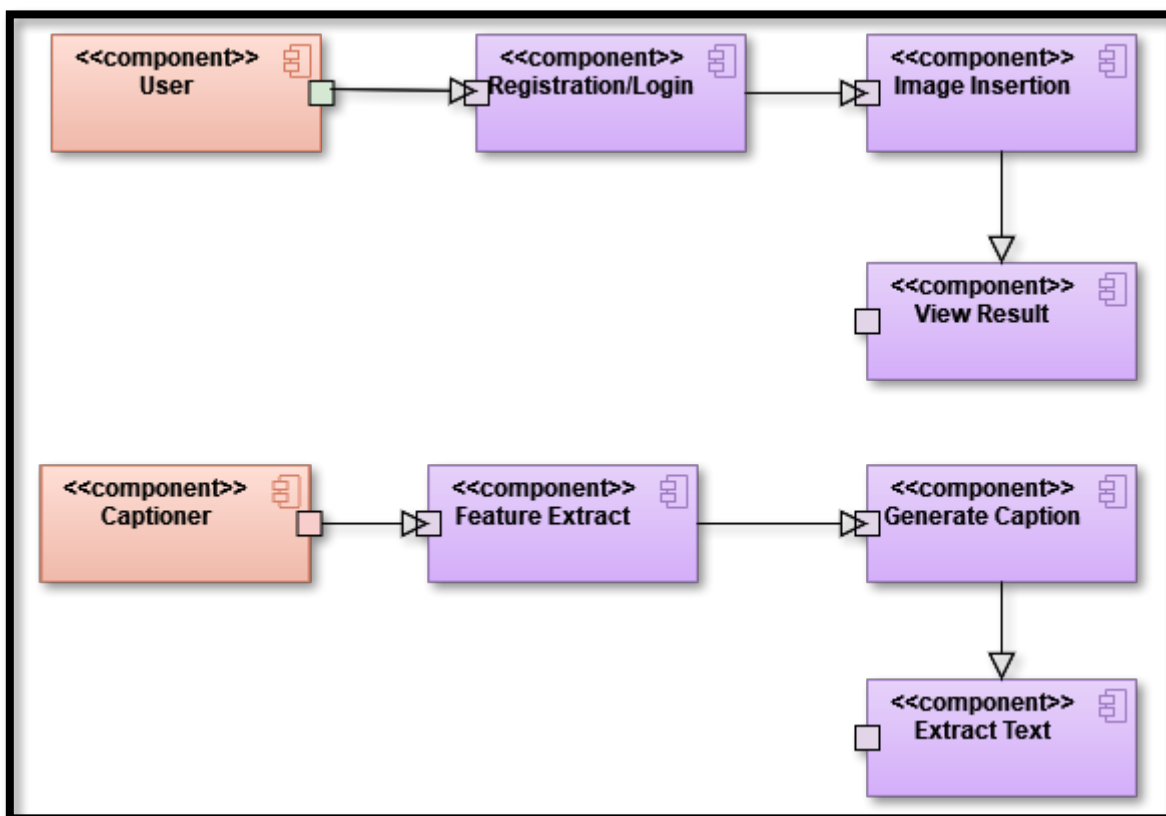


Figure 5.1 Component Diagram

#### Explanation

This diagram shows that only registered users can use the system. Users will login to the system by providing their details and are directed to the main screen of the application. Here users can upload the image from their device. The system will generate the relevant captions and extract the text from the image if any. These captions are displayed to the user and the user can convert them into speech by selecting text to speech option.

## 5.2. Architecture Diagram

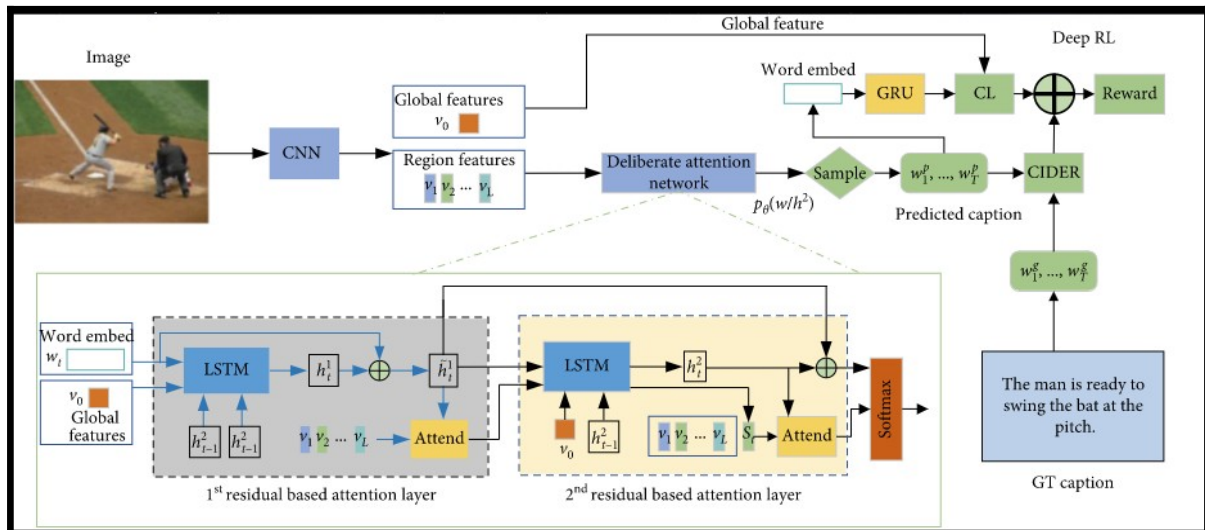


Figure 5.2 Architecture Diagram

### Explanation

This diagram shows the architecture of the image captioning system. The image is first passed through CNN model in which the features of the image are extracted. Then those extracted features are used as an input to the LSTM model in which the captions are generated.

### 5.3. Caption Generation Model

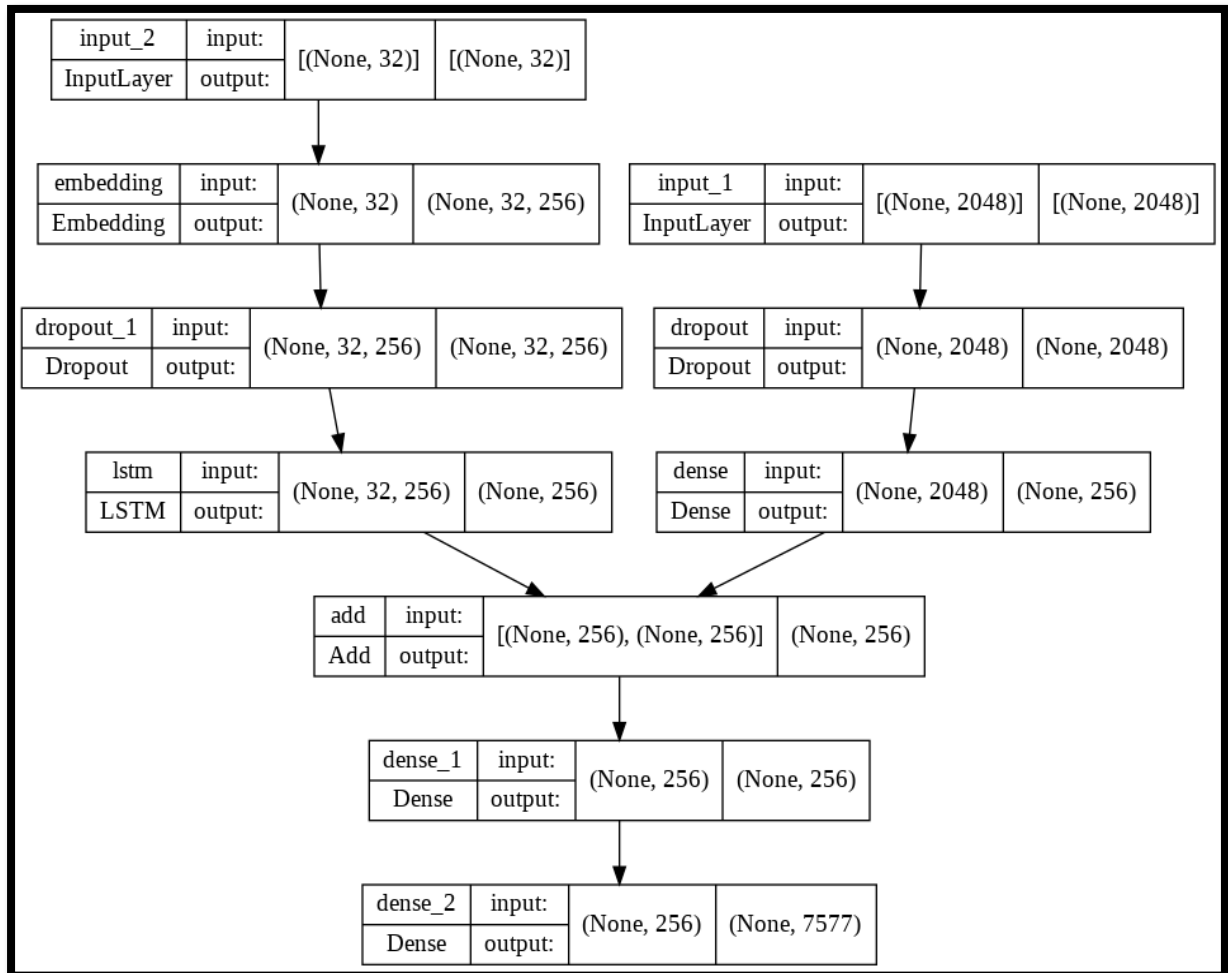
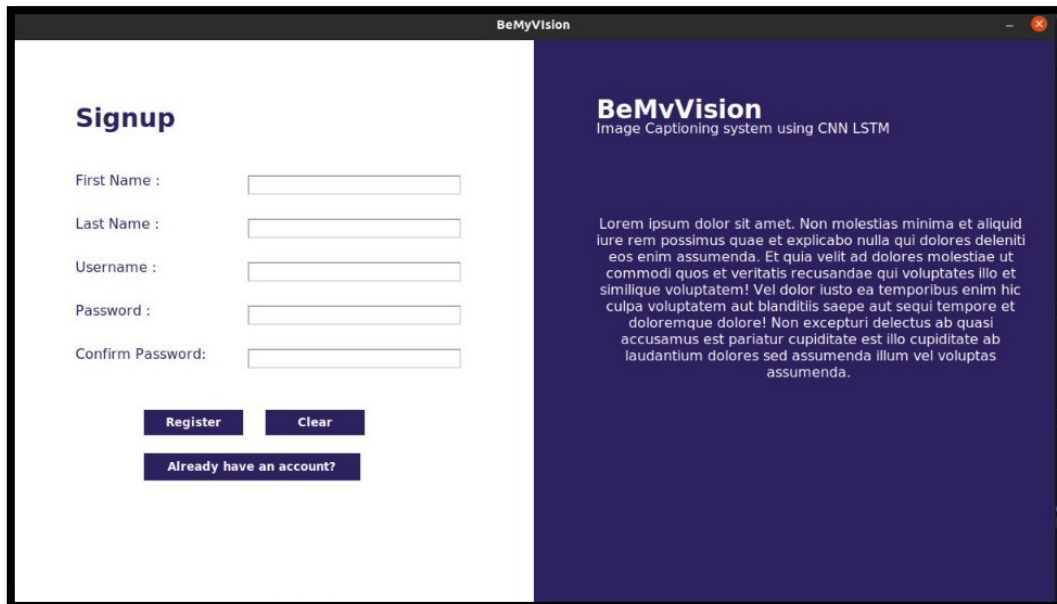


Figure 5.3 Caption Generation Model



## 5.4. User Interface

### Sign Up Screen:



The screenshot shows a web browser window titled "BeMyVision". The main content area is split into two columns. The left column, with a white background, is titled "Signup" and contains five input fields: "First Name :", "Last Name :", "Username :", "Password :", and "Confirm Password:". Below these fields are three buttons: "Register" and "Clear" (both in dark blue), and "Already have an account?" (in white on a dark blue background). The right column has a dark blue background and contains the "BeMyVision" logo and the tagline "Image Captioning system using CNN LSTM". Below this is a paragraph of placeholder text starting with "Lorem ipsum dolor sit amet..."

*Figure 5.4 Sign-up form*

### Explanation

For the registration process, user must enter a username, password and confirm password. The password and confirm password entered by the user should be same, otherwise, the user will not be able to register in the system. All the fields in the form should be filled in order to proceed further.

### Login Screen:

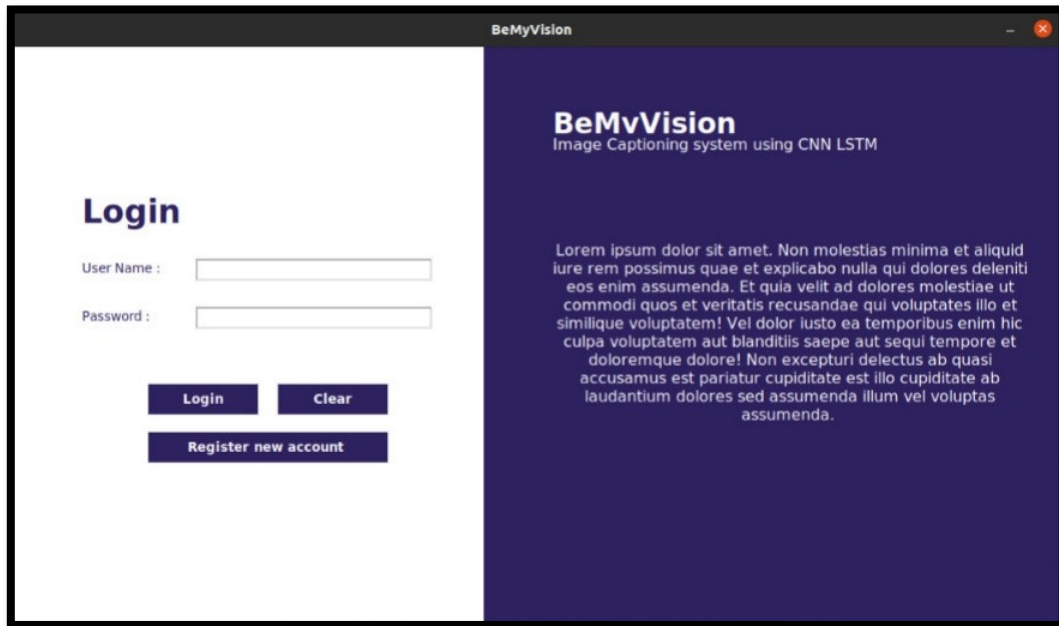


Figure 5.5 Login Form

**Explanation:**

The login form consists of username and password. The system will verify the details from the details stored in the user database. When the user is verified, he/she is logged into the system and transferred to the user panel.

**Main Screen:**

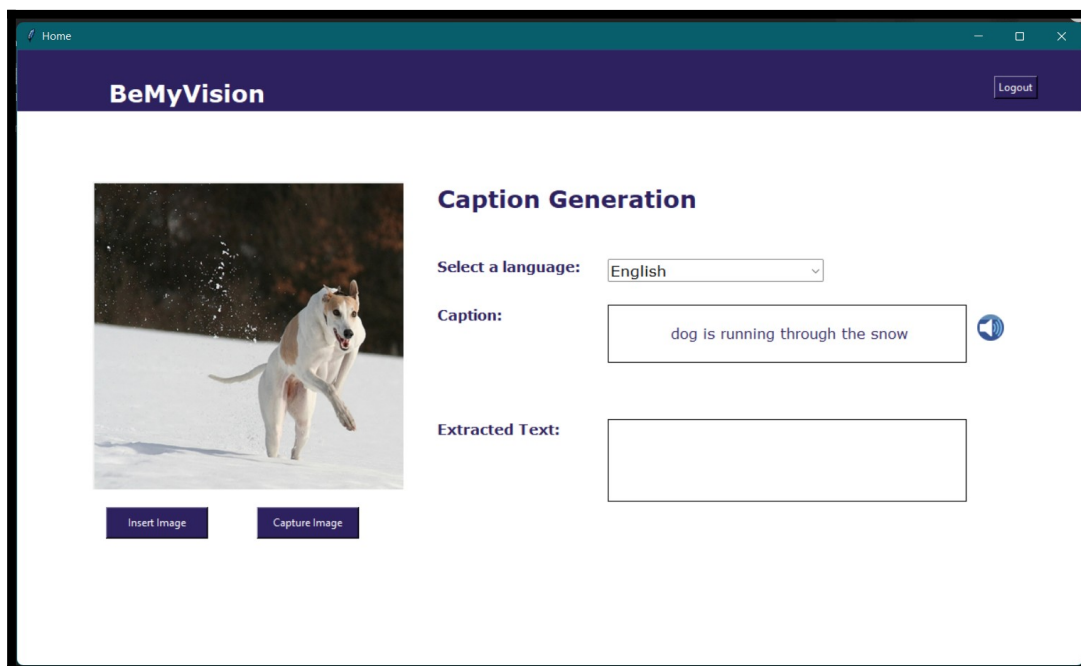


Figure 5.6 Main Screen

**Explanation:**

The user can insert an image as well as capture live image from the web camera. The system will take the image and generate its caption and display it in the text field. If any text, the system will extract the text with the help of OCR and display it in the text field below. By selecting any language from the menu, the caption is translated into the language and the audio file is generated in that particular language. The audio is played by clicking on the speaker icon.

## **Prediction Model**

### **6.1. System Architecture**

The CNN LSTM architecture involves using Convolutional Neural Network (CNN) layers for feature extraction on input data combined with LSTMs to support sequence prediction.

This architecture is mainly utilized for generating textual descriptions of images. Key is the use of a CNN that is pre-trained on a challenging image classification task that is re-purposed as a feature extractor for the caption generating problem.

A CNN LSTM can be defined by adding CNN layers on the front end followed by LSTM layers with a dense layer on the output.

It is helpful to think of this architecture as defining two sub-models: the CNN Model for feature extraction and the LSTM Model for interpreting the features across time steps.

### **6.2. Caption Generation Model**

In order to make an image caption generator model we have to combine both models CNN and LSTM. We can drive that:

Image Caption Generator Model (CNN-LSTM model) = CNN + LSTM.

- CNN- For extracting features from the image. A pre-trained model called Xception is used for this.
- LSTM- For generating a caption from the extracted information of the image.

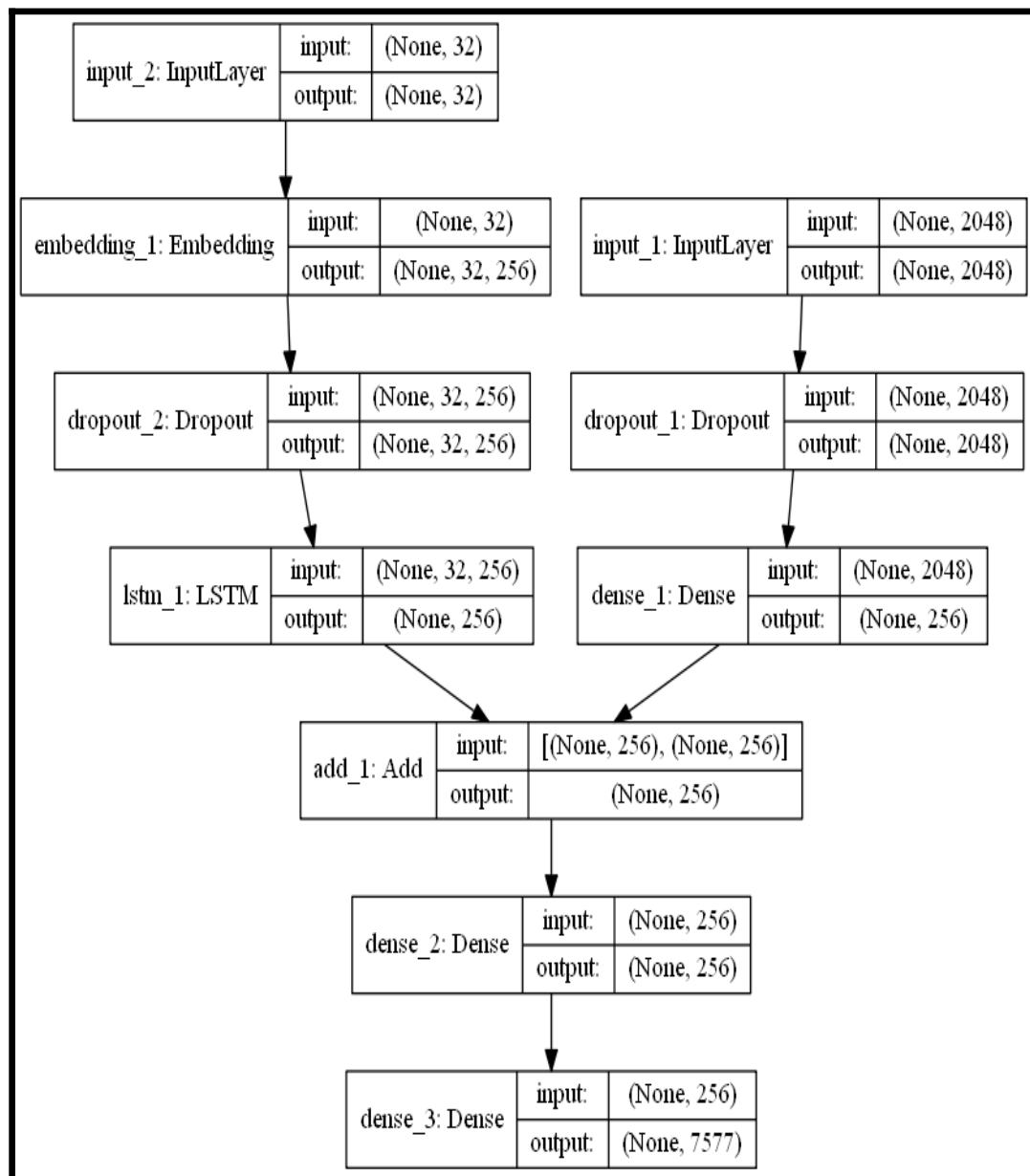


Figure 6.1 Image Caption Generation Model

The diagram above explains how the layers of CNN and LSTM are combined to become a complete caption generation model.

### Dataset

In this project we are using Flickr 8k [6] dataset. This dataset is best for image processing purposes. This dataset consists of 8000 images that are paired with 5 different relevant captions for each image. These captions provide clear descriptions of the salient entities and events. The images were manually selected to depict a variety of scenes and situations

The data will be split into training and testing datasets. 6000 images will be used for training as training dataset and 1000 images will be used for testing.



*Figure 6.2 Dataset*

### **Feature Extraction (CNN)**

We are using a pre-trained CNN to extract the features from photos in the dataset and store the features to file. The benefit is that the very large pre-trained models do not need to be loaded, held in memory, and used to process each photo while training the language model. Later, the feature extraction model and language model can be put back together for making predictions on new photos.

Xception is a convolutional neural network that is 71 layers deep. This pre-trained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images.

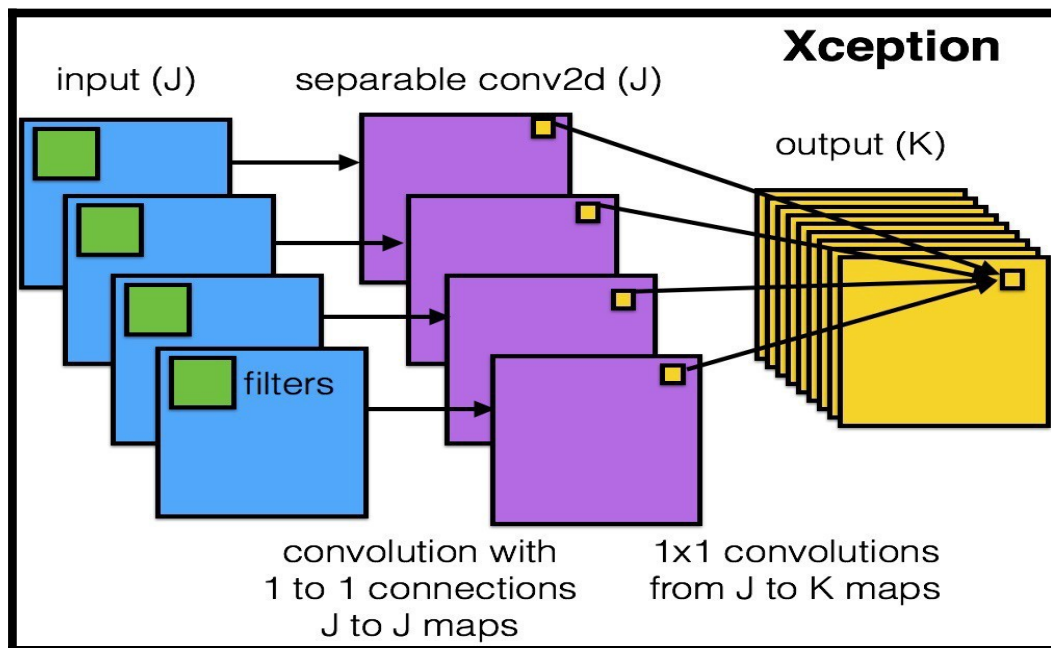


Figure 6.3 Xception Model

## LSTM

Being a type of RNN (recurrent neural network), LSTM (Long short-term memory) is capable of working with sequence prediction problems. It is mostly used for the next word prediction purposes, as in Google search our system is showing the next word based on the previous text. Throughout the processing of inputs, LSTM is used to carry out the relevant information and to discard non-relevant information.

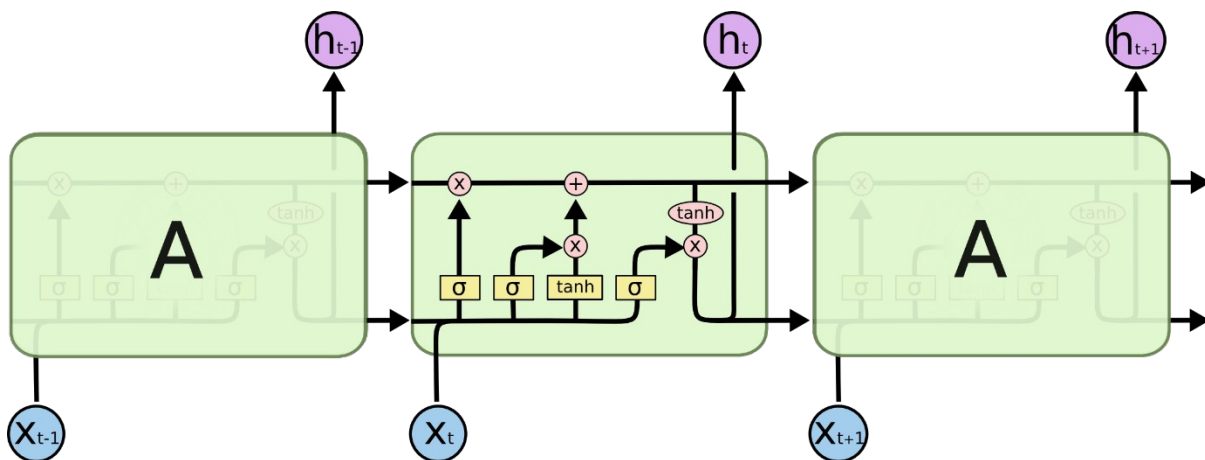


Figure 6.4 LSTM Architecture

The central role of an LSTM model is held by a memory cell known as a ‘cell state’ that maintains its state over time. The cell state is the horizontal line that runs through the top of the below diagram. It can be visualized as a conveyor belt through which information just flows, unchanged.

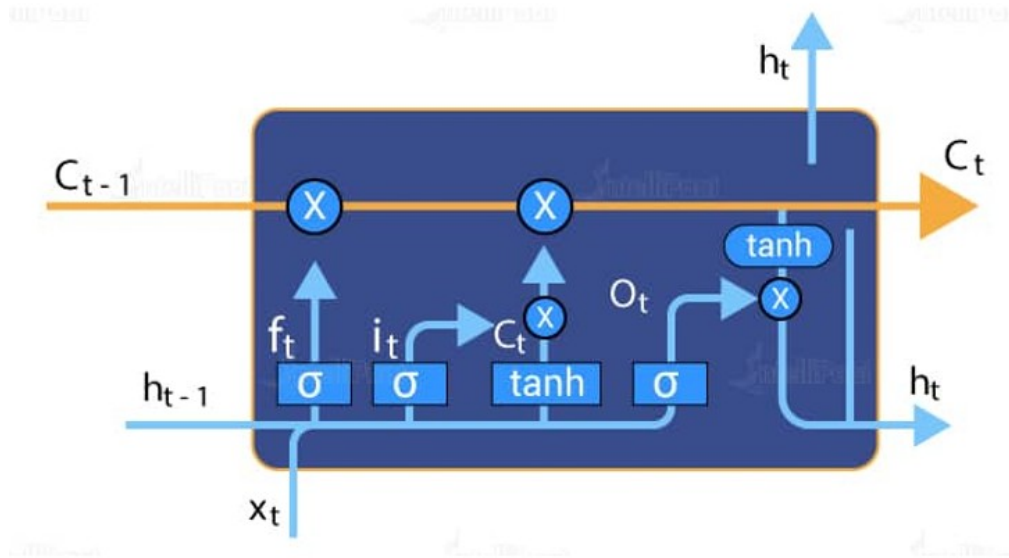


Figure 6.5 LSTM gates

Information can be added to or removed from the cell state in LSTM and is regulated by gates. These gates optionally let the information flow in and out of the cell. It contains a pointwise multiplication operation and a sigmoid neural net layer that assist the mechanism. The sigmoid layer gives out numbers between zero and one, where zero means ‘nothing should be let through,’ and one means ‘everything should be let through.’

### OCR Tesseract

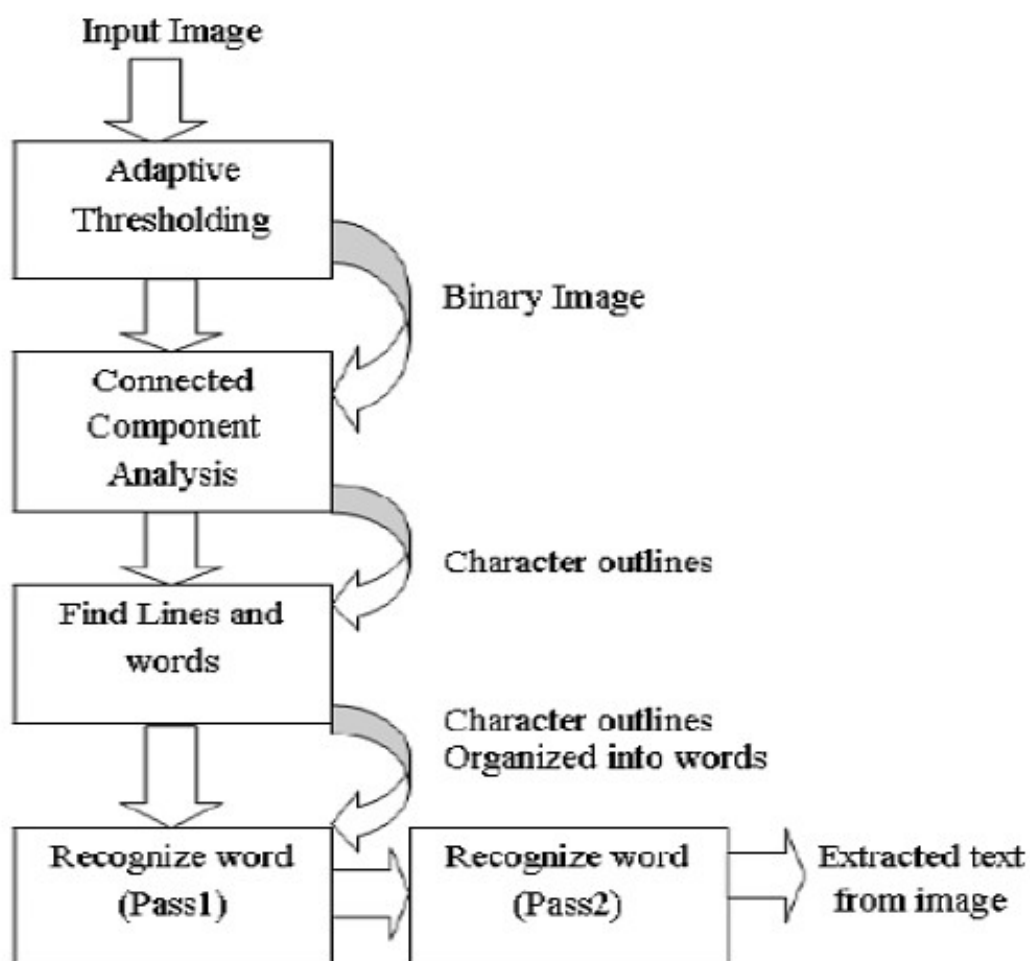
Employing OCR technology helps to reduce manual work of scanning documents, and as a result, a large number of documents can be processed with greater accuracy. For example, KYC authentication in the banking sector requires identifying information from documents such as passport, Pan Card, Aadhar Card, or Driving License and storing it in a database which an OCR quickly does.

Tesseract uses a defined set of techniques OCR processing. First, the image is converted into binary then a connected component analysis takes place, which stores the outlines of the

components. These outlines are then gathered as blobs. Next, blobs are structured into text lines which are further broken down into words based on the spacing of characters.

Once the words are detected, the next phase is about recognition. Recognition is a two-step process. In the first step, each word is tried to be recognized, and satisfactory words are passed to an adaptive classifier. When everything is parsed once, a second pass is taken to identify words that were not recognized in the first pass.

Ultimately, a final pass is taken to resolve fuzzy spaces and locate small cap text.



*Figure 6.6 OCR Process*

The given flowchart describes all the steps taken by the Tesseract OCR mentioned above to provide the output result.



## Testing and Evaluation

### 7.1. Verification:

Verification is the process of checking whether the software is able to achieve its goals and functions without any bugs. It is the process to make sure that the product developed is right or not. It also verifies whether the developed product fulfills all the product requirements that.

### Functional Testing:

Functional Testing is a type of black box testing where each part of the system is tested against functional requirements /specification.

- After giving the right credentials we can login to our system.
- Our system gives error messages when a user enters incorrect Login Credentials & when it is not registered.
- Our system successfully shows the record in the database of registered users.
- Our system successfully generates captions if an image is uploaded by the user.

### Sign up

*Table 6.1.1 Sign up Test Case*

Tested By	Ahtisham Hassan
Test type	Unit testing
Test case number	01
Test case name	Sign up
Test case description	This test case will check sign up form details
<b>Procedural Steps</b>	<b>Procedural Steps</b>
1	User will fill sign up form
2	All information must be entered correct as per

	validations.
3	User account will be created

## Login

*Table 7.1.1 Login Test Case*

Tested By	Maryam Khan
Test type	Unit testing
Test case number	02
Test case name	Login
Test case description	This test case will check login info
<b>Procedural Steps</b>	<b>Procedural Steps</b>
1	User will enter username and password
2	System will check if the account exists and validate the user info.
3	If account doesn't exist, then first have to create account

## Caption Generation

*Table 7.1.2 Caption Generation Test Case*

Tested By	Saba Syed
Test type	Unit testing
Test case number	03
Test case name	Caption generation
Test case description	This test case will allow user to upload image & get its caption.
<b>Procedural Steps</b>	<b>Procedural Steps</b>
1	User will login into the system.
2	User will upload the image.
3	System will then generate caption of that particular image.

## Translation

*Table 7.1.3 Translation Test Case*

Tested By	Saba Syed
Test type	Unit testing
Test case number	04
Test case name	Translation
Test case description	This test case will allow user to select a different language and translate the caption generated into that language.
<b>Procedural Steps</b>	<b>Procedural Steps</b>
1	User will login into the system.
2	User will select any language from the menu.
3	System will then translate the caption into selected language.

## 7.2. Software Testing

Testing is a very crucial yet important step of the software development process. It requires a complete evaluation of the software to make sure it meets the client’s requirements and goals. The most important purpose and goal of testing is to identify all the defects and errors in the software before the implementation takes place. If software defects are not addressed before deployment, they can have an adverse effect on the client’s business. E.g., resolving those issues would involve high costs.

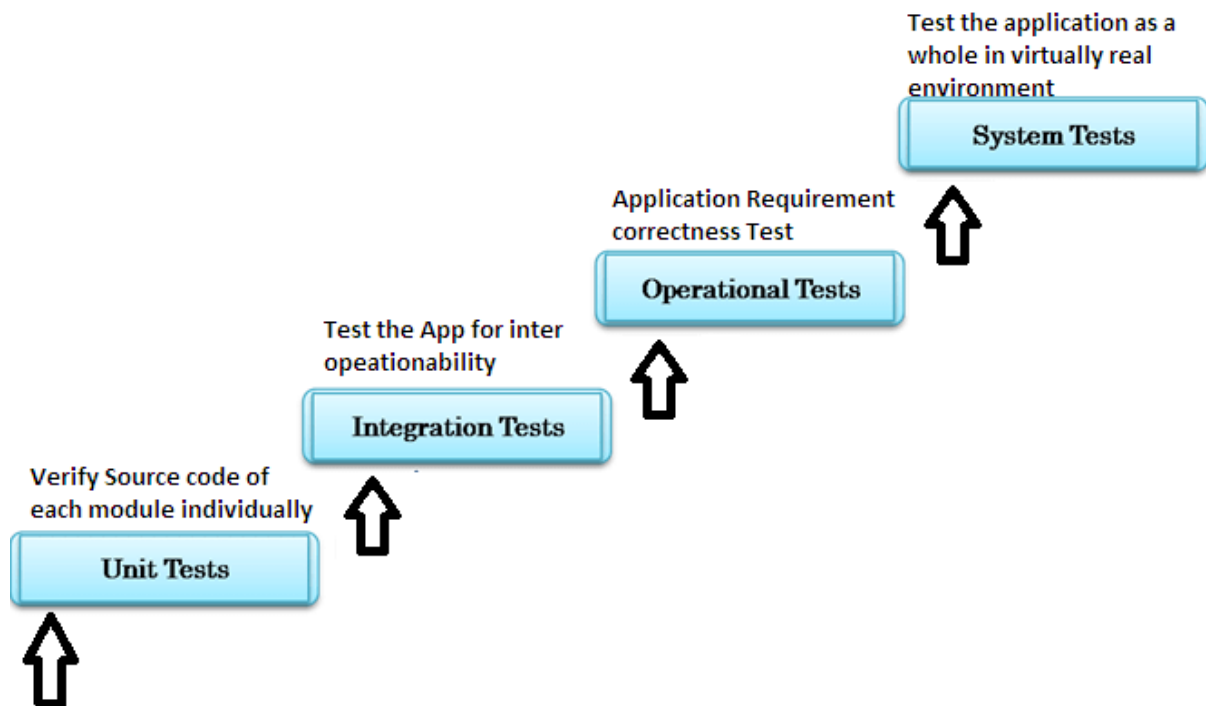


Figure 7.1 Software Testing Phases

## 7.3. GUI Testing

Our application is developed using Python Tkinter. The design is user-friendly and easy to understand and consists of decent theme and colors. GUI is the most important part of the system as it helps the user to interact with the application. Many test subjects were allowed to interact with the application so far and they didn’t find any bug or error.

#### **7.4. Usability Testing**

Our desktop application has a very simple and easy to understand interface and still attractive that makes the user keeps coming back. Key to usability is to make a decent, attractive yet easy to understand GUI. The application was given to many different people to test to performance and usability, their positive reviews ensured the usability criteria.

#### **7.5. System Performance Testing**

System Performance Testing is a testing technique in which the performance of the whole system is tested under different circumstances. Our application is efficient and responsive. It needs an internet connection to Login/Signup, as the user data is kept in a database server, and for caption generation. The application works well with both GPU and CPU; however, it works faster with a GPU as compared to when working on simple CPU. Other than that, it doesn't require any extra resources.

#### **7.6. Exception Handling**

Exception handling is used to deal with the cases when a function of the application is not working properly or when there's an input required but the field is submitted empty, by providing an alternate function in order to handle those particular errors. In short, exception handling is mainly done to avoid the application from crashing due to an error.

Following are some points to explain where and how exception handling is used in our application:

- If the user enters the wrong username or password while signing in, a dialogue box will be generated displaying the message "Username or Password is incorrect".
- If the user tries to register to a new account using a username that is already taken by another user, a dialogue box will be generated displaying the message "Username is already taken."
- While registering to a new account if the password doesn't match with the confirm password, a dialogue box will be generated displaying the message "Password and confirm password should be same".
- The system will check if the image file is empty or not before directing it to the caption generation function.

- If the audio button is selected before any caption is generated, a dialogue box will be generated displaying an error message.

### **7.7. Evaluation**

The project was completed on time. The tasks and modules are evaluated correctly.

### **7.8. Deployment**

Our project is deployed in a real time environment, and it works accurately. There is proper exchange of information between system and application, hence deployment was also done.

### **7.9. Maintenance**

Maintenance was done as we fulfilled all the objectives of the project and maintained all the detection and correction of errors in the system.

## **Conclusion**

In this project, we built a tool, using deep learning algorithms, which is capable of generating relevant captions for the image. The main and primary purpose is to help blind community understand images through audio.

CNN-LSTM is the basic architecture used in this system. The deep learning model generates a caption for the image and then our system translates that caption into the language that user has selected. After that the text is converted into speech with the help of gTTS, a python library to interface with Google Translate text-to-speech API.

Our system can be used by people all over the world as it provides multiple language translations of the captions. This language translation is possible with the help of GoogleTrans, a python library to interface with Google Translate API.

For training our models we used Flickr8k dataset. This dataset is best for image processing purposes. It has over 1000 object classes. This dataset consists of 8000 images that are paired with 5 different relevant captions for each image. These captions provide clear descriptions of the salient entities and events. The images were manually selected to depict a variety of scenes and situations



## **References**

- [1] Mrunmayee Patil, Ramesh Kagalkar, International Journal of Computer Applications  
Volume 118 – No. 3, May 2015
- [2] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In NIPS, 2014.
- [3] j. Johnson, A. Karpathy, and L. Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In CVPR, 2016.
- [4] D. Elliott, S. Frank, and E. Hasler. Multilingual Image Description with Neural Sequence Models. arXiv1510.04709, 2015.
- [5] T. Miyazaki and N. Shimizu. Cross-Lingual Image Caption Generation. In ACL, 2016
- [6] [www.kaggle.com/datasets/adityajn105/flickr8k](http://www.kaggle.com/datasets/adityajn105/flickr8k)