1. **INTRODUCTION**

**OVERVIEW**

This report discusses the result of the work done in development of “Caption Generator” on Machine Learning Platform. The project aims at the development of an application that identifies the action portrayed in the given image.

**PURPOSE**

You saw an image and your brain can easily tell what the image is about, but can a computer tell what the image is representing?

With the advancement in Deep learning techniques and availability of huge datasets and computer power, it is possible to build models that can generate captions for an image.

This project involves **Computer Vision** and **Natural Language Processing** concepts to recognize the context of an image and describe them in a natural language like English.

The objective of the project is to build a working model of Caption Generator by implementing CNN with LSTM.

The basic working of the application is that the features are extracted from the images using pre-trained **VGG16 model** and then fed to the **LSTM model** along with the captions to train.

The trained model is then capable of generating captions for any images that are fed to it.

1. **LITERATURE SURVEY**

**BACKGROUND AND MOTIVATION**

**Existing Systems**

* **Captionbot.ai**
  1. It is a product of Microsoft.
  2. It is an ML application that can understand the content of any image.
  3. When a person upload a photo, it is sent to Microsoft for image analysis to return a caption
  4. The application will not store or publish the images anywhere.
  5. It uses Computer Vision API and Emotion API
* **How-Old.net**
  1. It is also a product of Microsoft.
  2. It estimates the age of the person in the given image.
  3. The age is generated as a caption
* **TwinsOrNot.net**
  1. It identifies whether the image has twins or not

**Research Papers**

* Title: Automatic Description Generation from Images: A Survey of Models, Datasets, and Evaluation Measures
* Authors: Raffaella Bernardi, Ruket Cakici, Desmond Elliott, Aykut Erdem, Erkut Erdem, Nazli Ikizler-Cinbis, Frank Keller, Adrian Muscat, Barbara Plank
* Publication: Cornell University
* Summary: Here in this paper, the authors have described their project as a challenging problem as the model was not working properly with natural images that have recently received a huge amount of attraction from the computer vision and natural language processing communities. Also, they have classified the existing approaches based on how they conceptualized this problem. They have helped in reviewing the detailed description of existing models along with their advantages and disadvantages.
* Title: An Empirical Study of Language CNN for Image Captioning
* Authors: Jiuxiang Gu, Gang Wang, Jianfei Cai, Tsuhan Chen
* Publication: 2017 IEEE International Conference on Computer Vision
* Summary: In this paper, the effectiveness of their approach is validated on two datasets: Flickr30K and MS COCO. The extensive experimental results show that their method outperforms the vanilla recurrent neural network-based language models and is competitive with state-of-the-art methods.
* Title: A Comprehensive survey of Deep Learning for Image Captioning
* Authors: MD. Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, Hamid Laga
* Publication: Murdoch University, Australia
* Summary: In this paper, Although deep learning-based image captioning methods have achieved remarkable progress in recent years, a robust image captioning method that is able to generate high-quality captions for all images is yet to be achieved.
* Title: What is the Role of Recurrent Neural Networks (RNNs) in an Image Caption Generator?
* Authors: Marc Tanti, Albert Gatt, Kenneth P. Camilleri
* Publication: Cornell University
* Summary: In this paper, a recurrent neural network (RNN) is typically viewed as the primary ‘generation’ component. The authors suggest that the image features should be ‘injected’ into the RNN. They have viewed the RNN algorithm as only encoding the previously generated words. According to the authors, the RNN algorithm should only be used to encode linguistic features and that only the final representation should be ‘merged’ with the image features at a later stage. The paper compares these two architectures. As suggested RNNs are better viewed as encoders, rather than generators.

**OBJECTIVE**

The final goal of the project is:

A Machine Learning application that is used to portray the scenario depicted in the images uploaded by the user. This is possible as the application initially identifies the action portrayed in the given image.

The generated caption will describe about the image that will say what kind of actions is taking place in it.

**METHODOLOGY**

To implement the above goals, the following methodology needs to be followed:

1. Specifying the application and various components of the architecture.
2. Specifying the bindings between the various python packages and Machine Learning models.
3. Specifying the server ports between the modules of flask.
4. Analysis: Extracting the required data for analysis and then doing the analysis.

**ANALYSIS**

Based on analysis and literature survey regarding the present difficulties faced by the existing systems like Captionbot.ai, How-Old.net and TwinsOrNot.net as although Captionbot.ai does generate captions but it does not allow users to upload images as per their desires. How-Old.net does not give accurate predictions for all uploaded images and TwinsOrNot.net is a deprecated system.

Therefore, with the system of mine I am trying to generate captions for any images of any resolution once the users upload them.

**REQUIREMENTS ANALYSIS**:

**SOFTWARE REQUIREMENTS**:

Operating System: Windows 10 / Ubuntu

Front end: HTML, CSS, Bootstrap

Back end: Flask 1.x

Language: Python 3.7

Storage (Dataset): Google Drive

IDE: Jupyter Notebook

Cloud Deployment: Heroku Deployment

Other Technologies used: Git and GitHub

**HARDWARE REQUIREMENTS:**

RAM: 8GB and above

Hard disk: 120GB and above

Processor: Intel i3 and above

**FUNCTIONALITIES**:

* **User Perspective**
  + User launches the application.
  + User uploads the images for which the captions are to be generated.
  + User clicks on the submit button.
  + Within a matter of seconds, the ML model recognizes and process the content of an image.
  + The result will be displayed on the page.
* **Model Development and Deployment**
  + Data Gathering
    - Images for training the model
    - Corpus for Image captions
  + Pre-processing
    - Corpus – Removal of stop words, punctuation marks, digits
    - Generating the bag of words
    - Mapping of images with the Corpus
  + Feature extraction from images
  + Object identification
  + Model Generation using CNN
  + Building LSTM (Long Short Term Memory) model.
  + Validating the Model
  + Deploy the ML model in the web application

**TOOLS AND TECHNOLOGIES**:

**APPLICATION DEVELOPMENT TECHNOLOGIES**:

This application is built using Python, Flask, HTML, CSS, and Bootstrap.

**Python** is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace.

**Flask** is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

**HTML (Hypertext Markup Language)** is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

**CSS (Cascading Style Sheets)** is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

**Bootstrap** is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.

**INTEGRATION TOOLS**:

**GIT AND GITHUB**:

Git is a distributed version-control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed, data integrity and support for distributed, non-linear workflows.

GitHub is a web based hoisting service for version control using Git. It is mostly used for computer code. It offers all of the distributed version control and source code management functionality of Git as adding its own features.

**HEROKU**:

**Heroku** is a container-based cloud Platform as a Service (PaaS). Developers use **Heroku** to deploy, manage, and scale modern apps. Our platform is elegant, flexible, and easy to use, offering developers the simplest path to getting their apps to market.

**DESIGN**

**DFD**

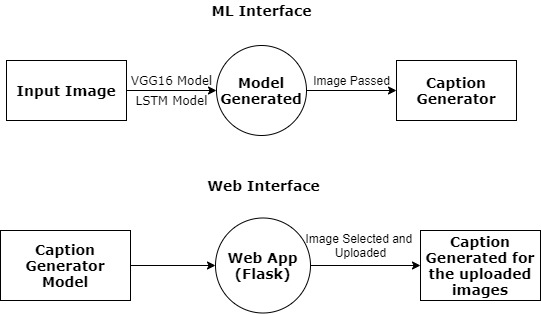


Figure 3.1.1 DFD Level 0

Our project consists of only one fold:

1. Visually impaired users: Users who would like to generate captions for the images that they had captured.

**PROCESS FLOW DIAGRAM**

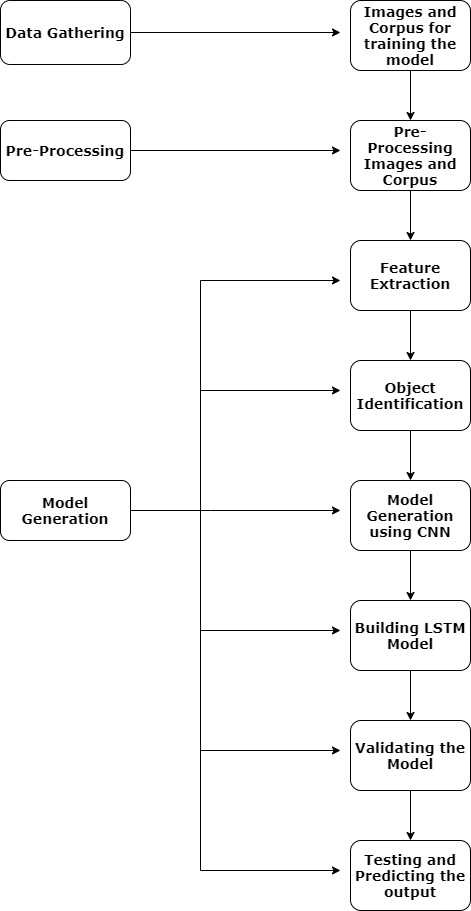


Figure 3.1.2 Use Case Diagram

**SCREEN SHOTS**

**HOME SCREEN**

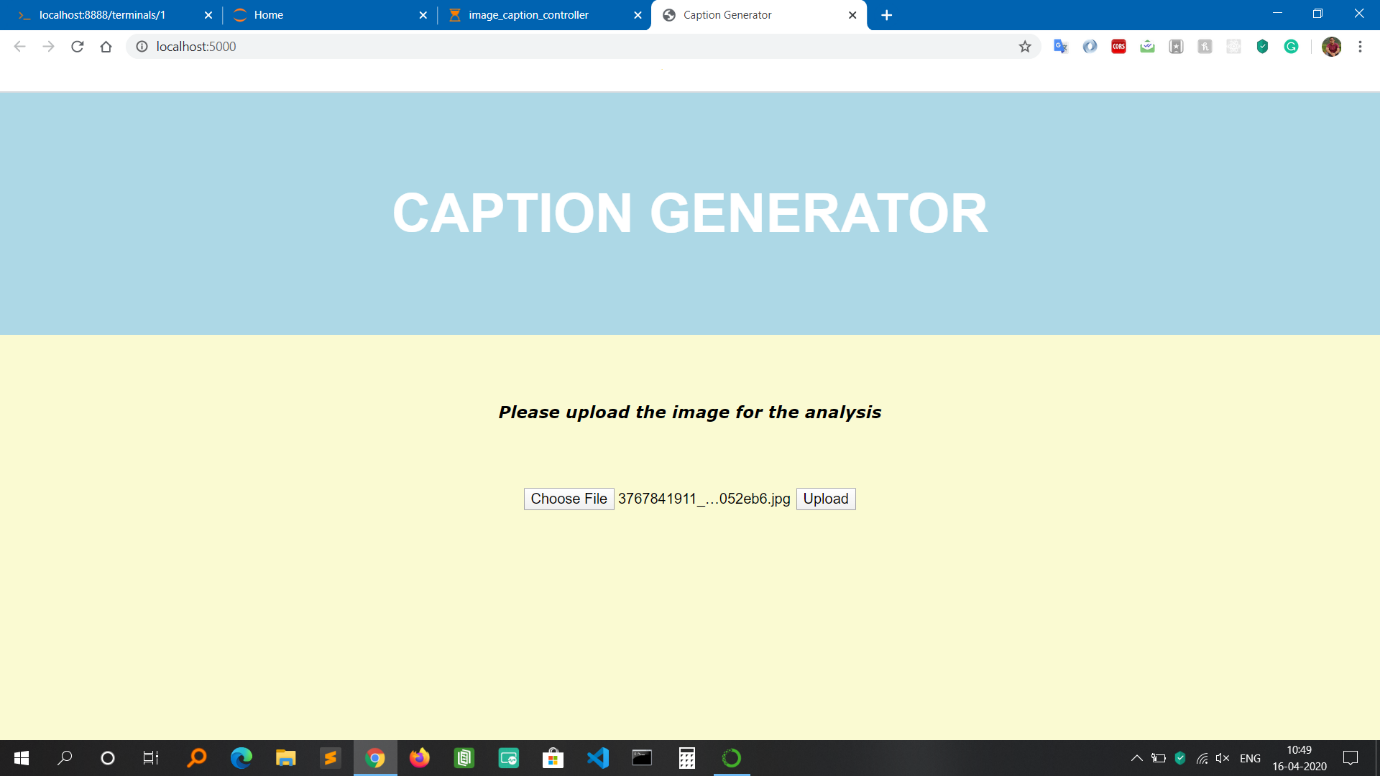


Figure 4.1.1 Home Screen

**RESULT PAGE**

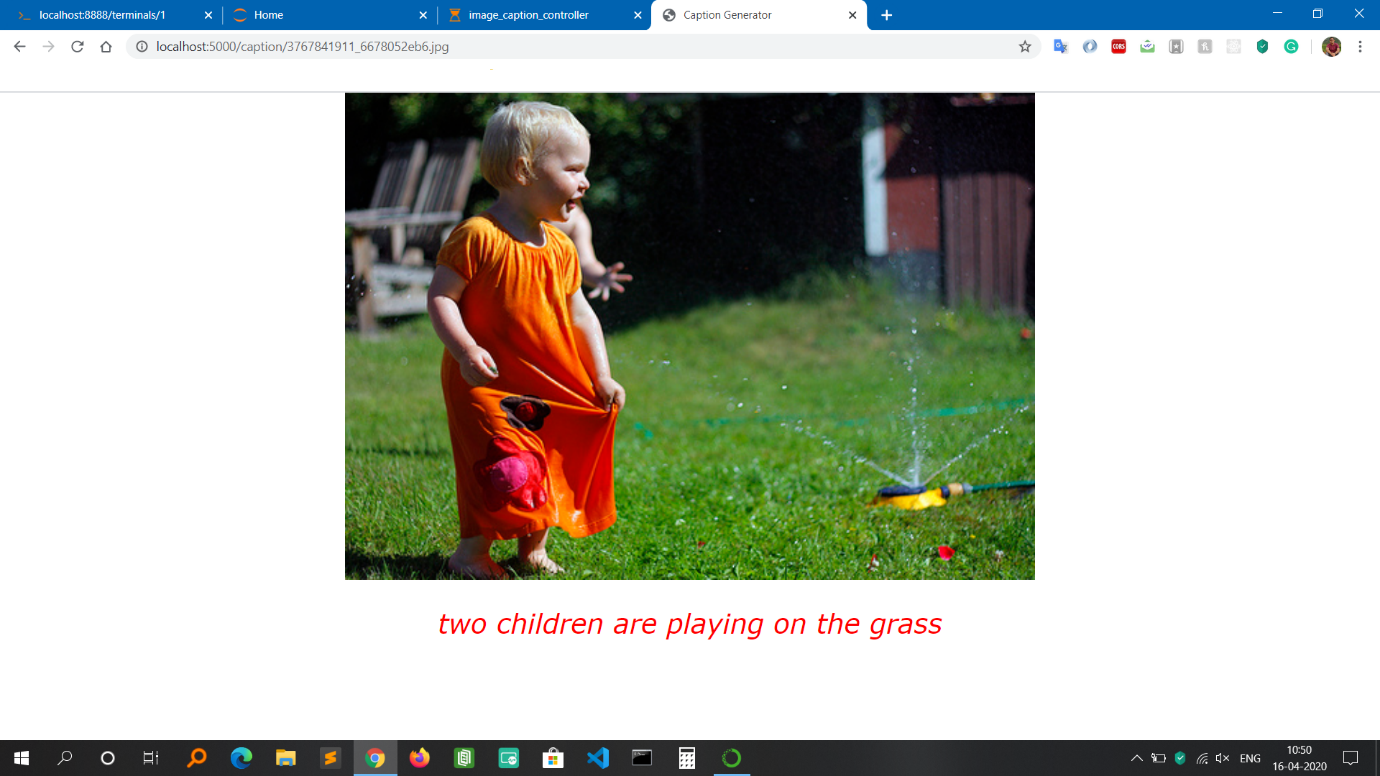


Figure 4.1.2 Result Page Screen

**TESTING**

Table 5.1 Test Case

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | PES\_001 | **Test Case Description** | | Test the Functionality of uploading the image for analysis | | | | | | |
| **Created By** | |  | **Reviewed By** | |  | | **Version** | | 1 | | |
|  |  |  |  |  |  |  |  |  |  |  |
| **QA Tester’s Log** | |  | | | |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Tester's Name** | |  | **Date Tested** | | 16-April-2020 | | **Test Case (Pass/Fail/Not Executed)** | | Pass | | |
|  |  |  |  |  |  |  |  |  |  |  |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | | |
| 1 | Access to Chrome Browser | | |  | 1 | Choose file = Upload images of any format like jpg, jpeg, png | | | | | |
| 2 |  | | |  | 2 |  | | | | | |
| 3 |  | | |  | 3 |  | | | | | |
| 4 |  | | |  | 4 |  | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |
| **Test Scenario** | Verify on ability to upload the images of any format for analysis after selecting the desired image | | | | | |  |  |  |  | |
|  |  |  |  |  |  |  |  |  |  |  |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | **Pass / Fail / Not executed / Suspended** | | | |
|
| 1 | Navigate to <http://localhost:5000/> | | Caption generator homepage should open | | Successfully redirected | | | Pass | | | |
| 2 | Click on Choose File to select the image for uploading | | File Explorer should open for selecting the image | | File Explorer Window successfully opened | | | Pass | | | |
| 3 | Select the desired image | | Should display the name of the image selected in the home page of application | | Image name is being displayed on the home page of application | | | Pass | | | |
| 4 | Click on Upload button to upload the image for analysis | | ML processing should begin in the background and should redirect to the result page for displaying the caption with the respective image | | Page redirected and caption generated for the uploaded image. | | | Pass | | | |

Table 5.2 Test Cases

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | PES\_002 | **Test Case Description** | | Test the Functionality of uploading the image for analysis without selecting any image | | | | | | |
| **Created By** | |  | **Reviewed By** | |  | | **Version** | | 1 | | |
|  |  |  |  |  |  |  |  |  |  |  |
| **QA Tester’s Log** | |  | | | |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Tester's Name** | |  | **Date Tested** | | 16-April-2020 | | **Test Case (Pass/Fail/Not Executed)** | | Fail | | |
|  |  |  |  |  |  |  |  |  |  |  |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | | |
| 1 | Access to Chrome Browser | | |  | 1 | Click the upload button | | | | | |
| 2 |  | | |  | 2 |  | | | | | |
| 3 |  | | |  | 3 |  | | | | | |
| 4 |  | | |  | 4 |  | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |
| **Test Scenario** | Verify on whether the home page gets redirected to result page without selecting any image | | | | | |  |  |  |  | |
|  |  |  |  |  |  |  |  |  |  |  |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | **Pass / Fail / Not executed / Suspended** | | | |
|
| 1 | Navigate to <http://localhost:5000/> | | Caption generator homepage should open | | Successfully redirected | | | Pass | | | |
| 2 | Click on Upload button to upload the image for analysis | | Validator should display message saying ‘Please select a file’ as no image is selected for uploading | | Page not redirected and Validator displays message saying ‘Please select a file’ | | | Fail | | | |

Table 5.3 Test Case

****

Table 5.4 Test Case

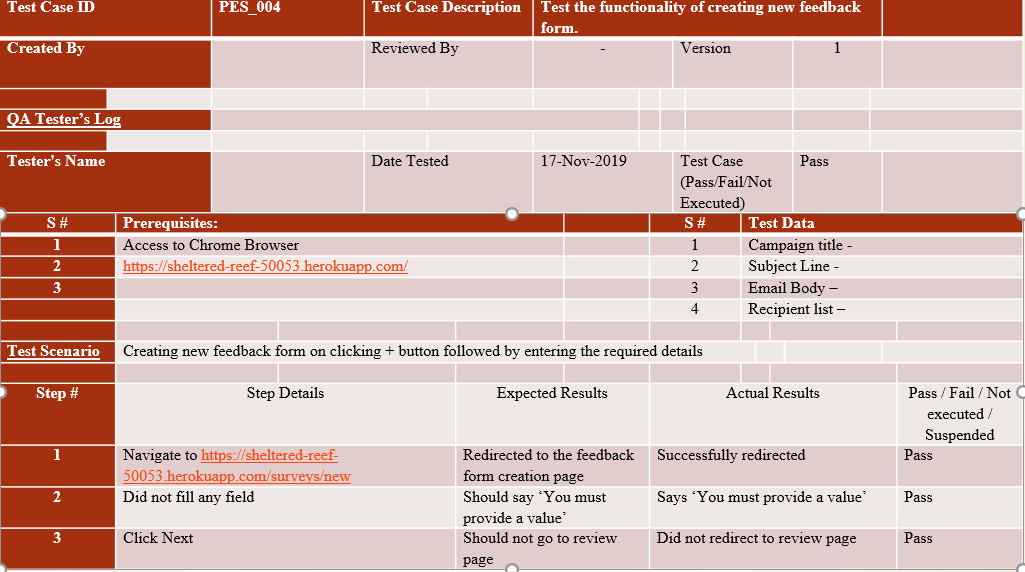
****

Table 5.5 Test Case

****

Table 5.6 Test Case



Table 5.7 Test Case

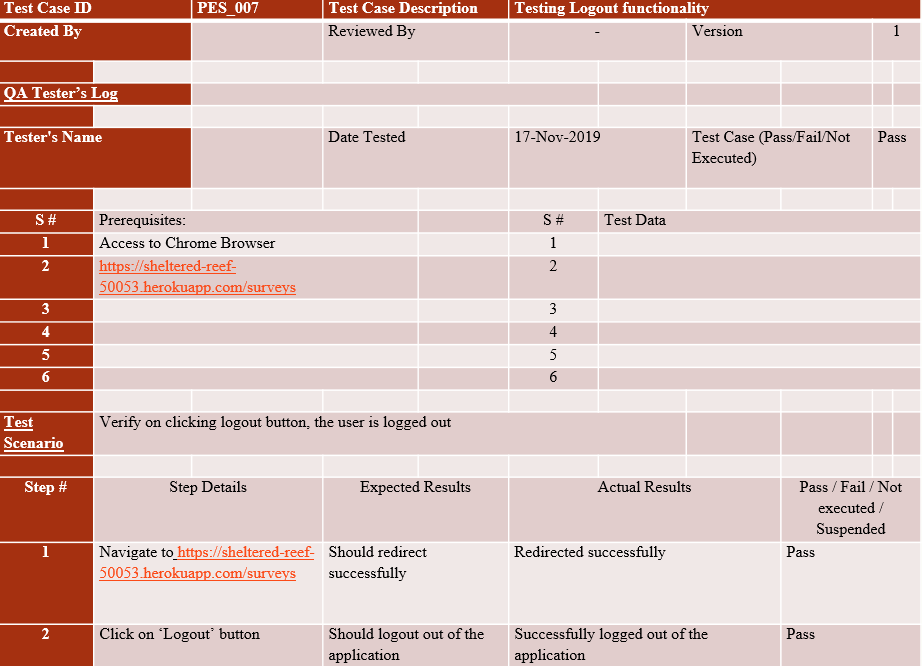
****

Table 5.7 Test case

**CONCLUSION**

1. The objective of the project was to identify the action portrayed in the given image. The generated caption will describe the image that will say what kind of actions is taking place in it.
2. This has been solved with my application as it provides a service to users that helps in generating captions for any images of their choice once they upload it in my application.
3. Besides, the whole application has been deployed on Heroku platform. So in the future, if the user requests for any changes, it can be easily done through git version control.

**FUTURE ENHANCEMENT**

1. I can build a customized mobile app that will make the user more convenient to use.
2. In the future, with a little more enhancement of algorithms, I will be able to achieve a higher accuracy that will help in better prediction of images.

**BIBLIOGRAPHY**

1. Book on **Deep Learning for Natural Language Processing** by **Jason Brownlee**
2. <https://www.kaggle.com/>

1. <https://www.medium.com/>
2. <https://www.towardsdatascience.com/>
3. https://www.machinelearningmastery.com
4. https://www.sciencedirect.com
5. https://www.researchgate.net