Memes Generator

Contents

[1 Preface 1](#_Toc48168880)

[2 Introduction 1](#_Toc48168881)

[What is the Meme Generator? 1](#_Toc48168882)

[3 Prerequisites 2](#_Toc48168883)

[4 Application Architecture 2](#_Toc48168884)

[5 Python Implementation 3](#_Toc48168885)

[5.1 Project structure 3](#_Toc48168886)

[5.2 Code Explanation 4](#_Toc48168887)

[5.3 Output 5](#_Toc48168888)

[5.3.1 Home Page: 5](#_Toc48168889)

[5.3.2 Text Input Page: 6](#_Toc48168890)

[5.3.3 Output Page: 6](#_Toc48168891)

# Preface

This document is intended towards helping all the data scientists out there. It is a step by step guide for creating a memes generator. In this case, a memes generator right from scratch and can be deployed in any cloud platform. This document takes a simple example of a memes generating process from scratch and then its deployment to a cloud environment.

# Introduction

## What is the Meme Generator?

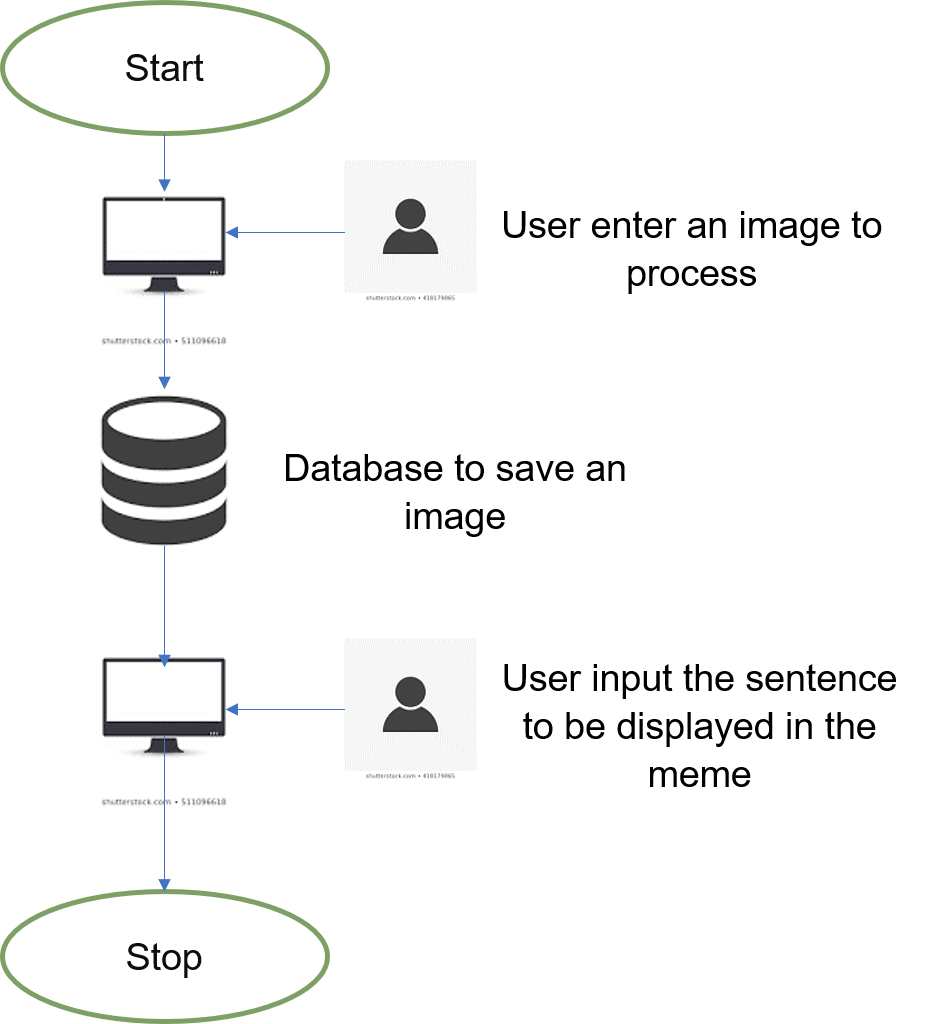
It's a free online image maker that allows you to add custom resizable text to images. It operates in HTML5 canvas, so your images are created instantly on your own device. Most commonly, people use the generator to add text captions to established [memes](https://en.wikipedia.org/wiki/Meme), so technically it's more of a meme "captioner" than a meme maker. However, you can also upload your own images as templates.

# Prerequisites

The things needed before we start building a python based memes generator are:

* Python installed.
* A Python IDE (Integrated Development Environment): like PyCharm, Spyder, or any other IDE of choice (Explained Later)
* Flask Installed. (A simple command: pip install flask)
* Basic understanding of Python and HTML.
* Basic understanding of Git (download Git CLI from https://gitforwindows.org/ )

# Application Architecture



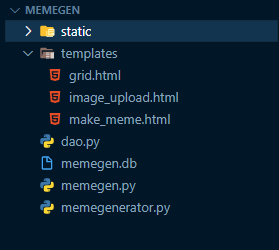
# Python Implementation

## Project structure

Note: VSCode has been used as an IDE for this documentation

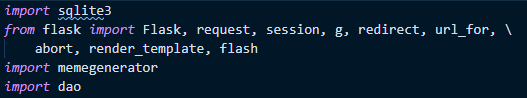
1. Let’s create a folder called ‘memegenerator’ on our local machines.
2. Inside that folder, let’s create two more folders called ‘static’ and ‘templates’ to hold
3. the code for the UI of our application. Inside ‘static’, let’s create a folder called ‘css’
4. for keeping the stylesheets for our UI.
5. Let’s create a file called ‘memegen.py’ inside the ‘memegenerator’ folder.
6. Inside the folder ‘templates’, create three HTML files called: ‘base.html’,’index.html’, and ‘results.html’. The files are attached here for reference.

* image\_upload.html→Takes image as input.
* make\_meme.html→ Takes the sentenceas input which are to be shown in meme.
* grid.html→ Page to show the meme for image and input sentences.



## Code Explanation

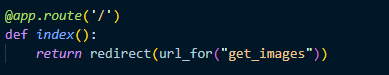
1. Import the necessary libraries:



1. Initialize the flask app

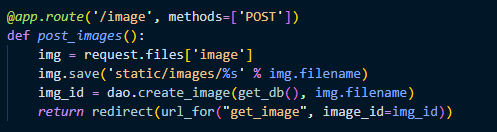


1. Creating the routes to redirect the control inside the application itself. Based on the route path, the control gets transferred inside the application.

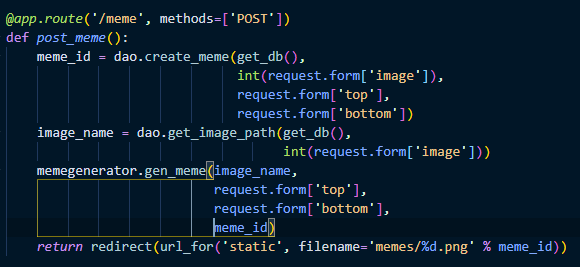


1. Now let’s understand the functions.

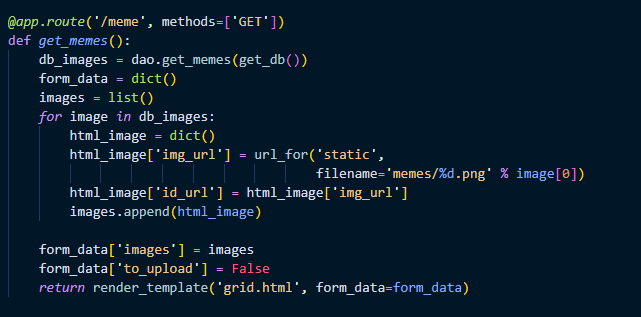
* If the HTTP request method is POST (which is defined in image\_upload.html at form submit action), then the image will be stored in database and user will be directed to.



* In the make\_meme.html the user is asked to input the meme text for bottom and top of the meme.

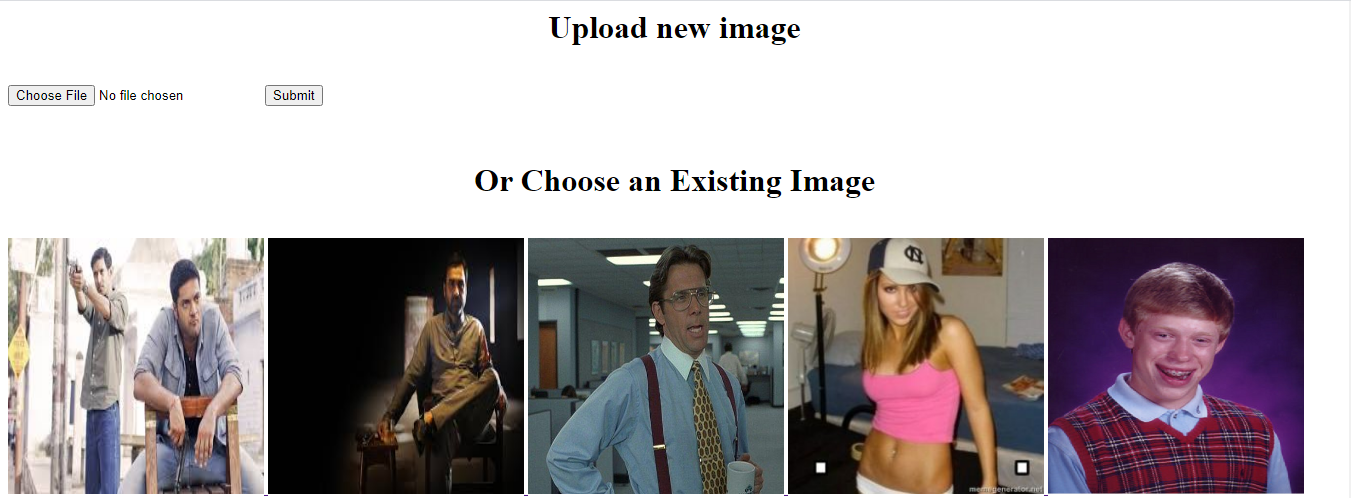


* The meme is generated using the memegenerator.py file and the output meme is shown to user.



## Output

### Home Page:



### Text Input Page:



### Output Page:

