MemeIt – 0.0.2

Talent Boost Final Task

General

The task is to extend the previous application by making it possible to display both local memes and memes created by other similar applications. In order for that to happen, you have to use the "**MemeIt Platform Service**" as a registry for different "MemeIt" instances (below they are referred to as domains). In this service you can: **query all**, **register** or **deregister** an instance.

MemeIt Platform Service

Note: This service is no more than a simple registry for domains – think of it as a map with addresses of all registered "MemeIt" instances.

The service has the following REST API:

GET /domain

Description: Returns all registered domains
Response: List of all domain objects each having **name** and **address**.
Response:

```
[
  { "name": "Programming", "address": "http://1.1.1.1:8080" },
  { "name": "Stranger Things", "address": "http://1.1.1.2:8080" },
  { "name": "GoT", "address": "http://1.1.1.3:8080" },
  …
]
```

POST /domain/register

Description: Adds a domain in the registry
Body: A single object containing a human-readable **name** (describing the topic for the majority of your memes) and an **address** of the backend endpoint which later on can be queried.
Body:

```
{
  "name": "Programming",
  "address": "http://1.1.1.1:8080"
}
```

Response: unique identifier as **integer**.
Status codes returned are: 200 – OK, 400 – Bad Request, 409 – Conflict – name is taken.

DELETE /domain/deregister/:id?

Description: Removes a domain from register by specifying id of the target domain.

```
{
  "id": Integer
}
```

Example:

```
/domain/deregister/1
```

Response: Only **status code** is returned (200 – OK, 400 – Bad Request, 404 – Not Found)

MemeIt Domain Application

Each application ("MemeIt" instance) registered to this service should have the following REST API signature:

GET /meme

Description: Returns all memes in the application

Response: List of all memes each having **title** and **image source** (a URL used to access the image).

Response Example:

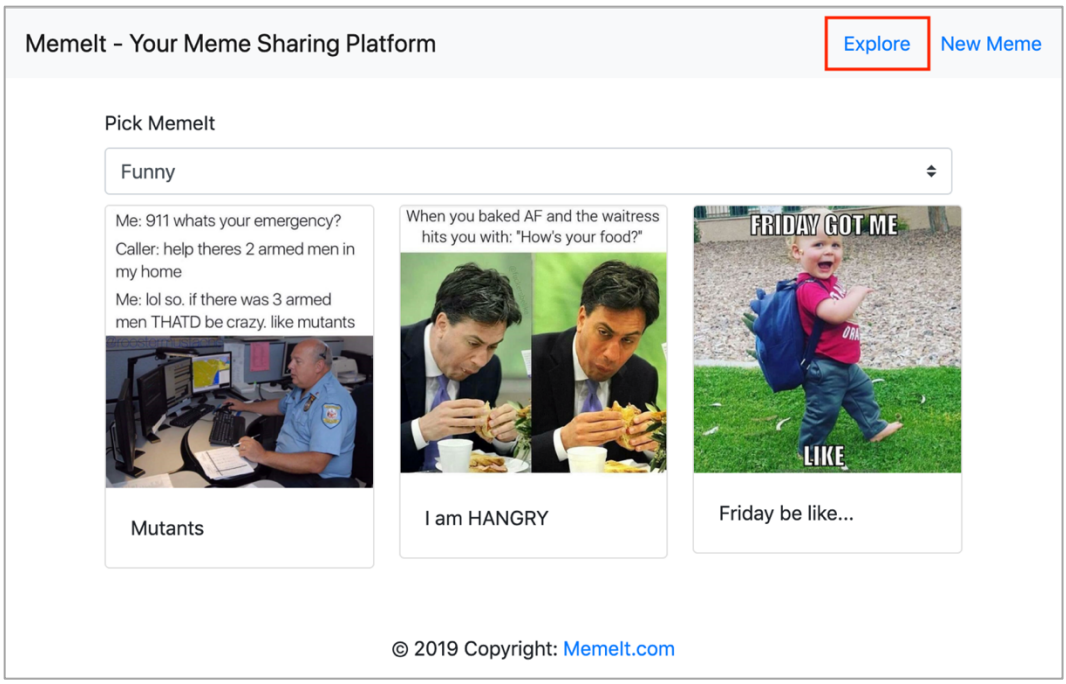
```
[
  {
    "title": "Ain't nobody got time for that",
    "image": "http://my-server-address/static/image-934763.jpg"
  },
  {
    "title": "Brace yourself…",
    "image": " http://my-server-address/static/image-13214.jpg "
  },
  …
]
```

Main Application Components

Explore Screen (20pts)

It should contain:

- all available domains in a dropdown
- all memes per selected domain



Register Domain in Platform (5pts)

Whenever you **start the application** you should try to **register** to the "MemeIt Platform Service". The location of the server will be given to you during the exam. The name with which you register to the domain is not important (it could be hardcoded) – just make sure it is appropriate.

Deregister from Platform (5pts)

Whenever you **exit the application** make sure that you **deregister** from the “MemeIt Platform Service”.

MemeIt – 0.0.3

Talent Boost Final Task

General

The task is to create a web app in which you upload and share memes. The app will be accessed by multiple users simultaneously.

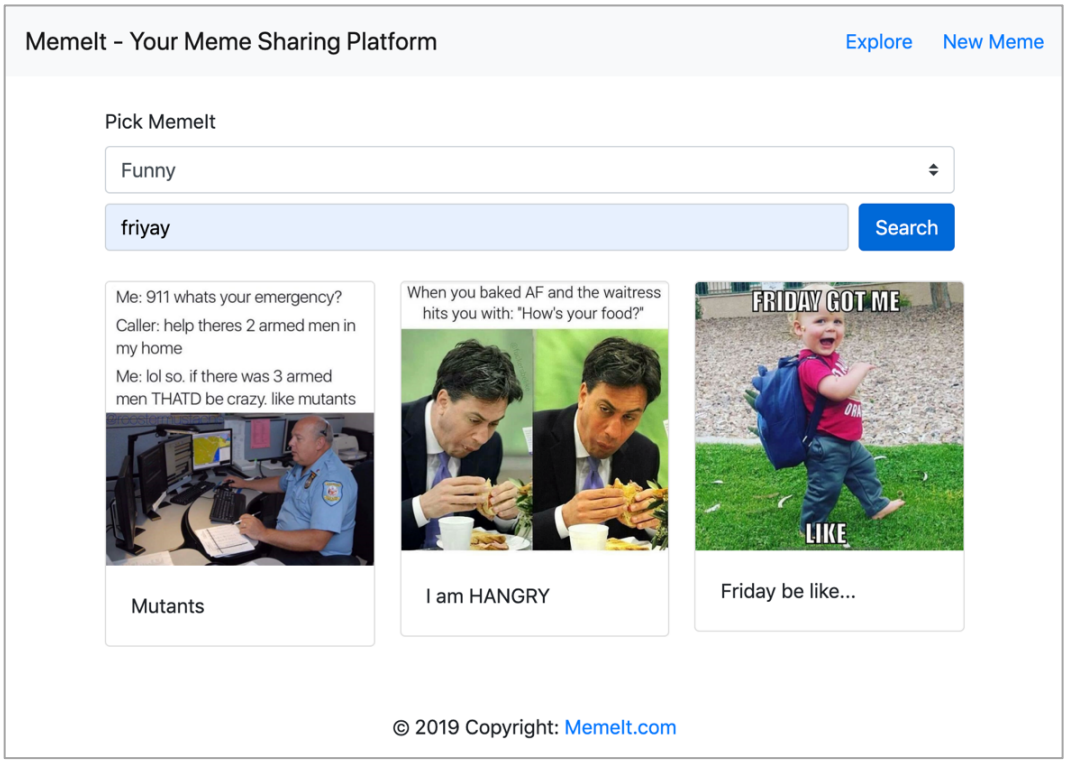
Main Application Components

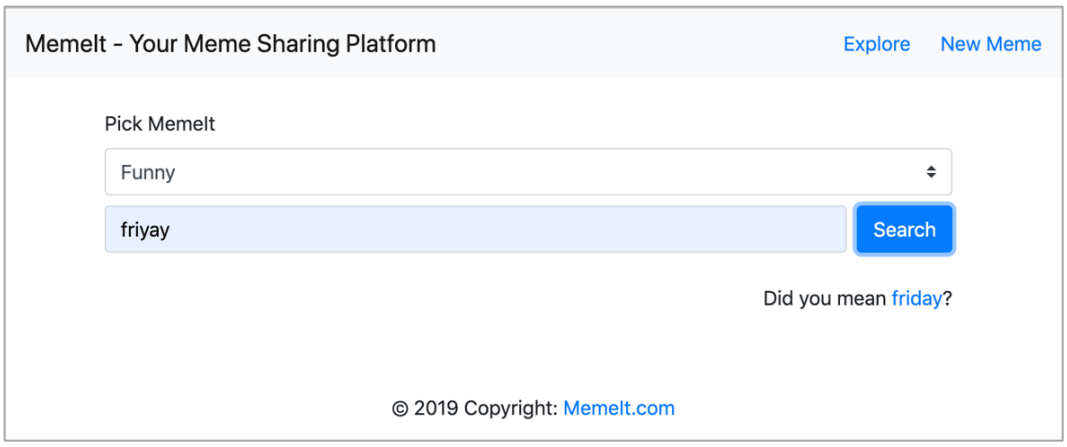
*Bonus

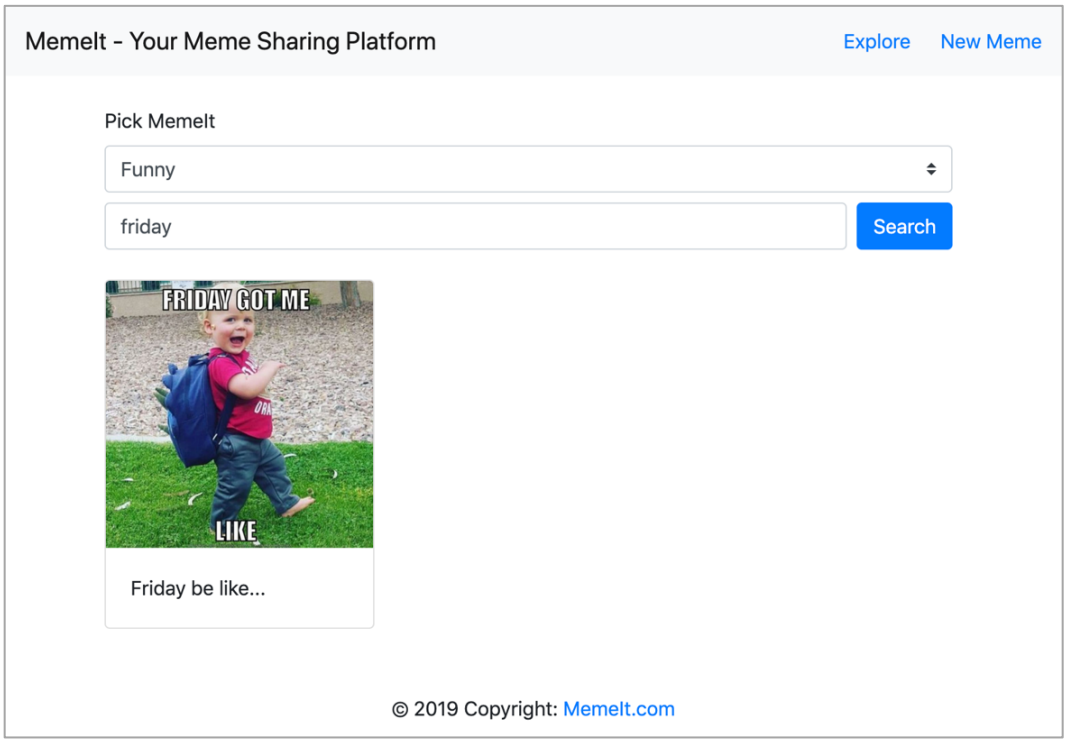
Explore Page 15 pts

Explore screen should contain:

- advanced search
  - the search should check whether the searched term is a part of any meme's title
  - if no match is found the application should be able to suggest to the user a new search term based on the similarity of the current one







Hints:

1. Add a form with input field and submit button and bind to it's submit event ("*ngSubmit*")
2. On form submit make a call to the backend containing the searched term
3. If there are any results return them – if not use some sort of algorithm (or some third-party library) to *find the closest match* that will return some results (memes) and return it.
4. Display the final result in the UI

*Bonus

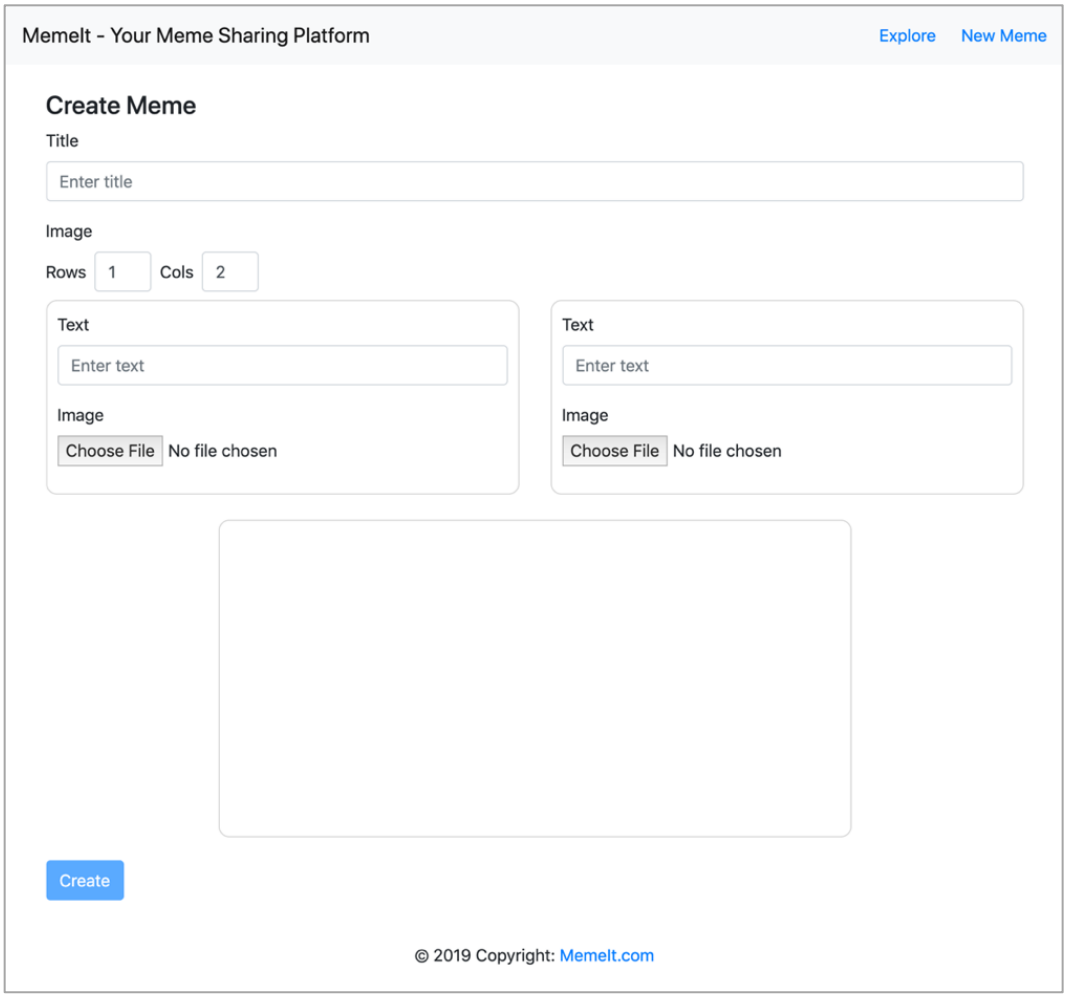
Add a way for crafting a meme by including more than one image (each image should have text defined as well).

Create Meme Page (20pts)

It should display a web form for meme creation.
A meme has the following fields:

- Title
- Number of rows – R – 1min, 2 max
- Number of columns – C – 1 min, 2 max
- Boxes - count is R * C. Each box has following fields:
    - Background Image
    - Text

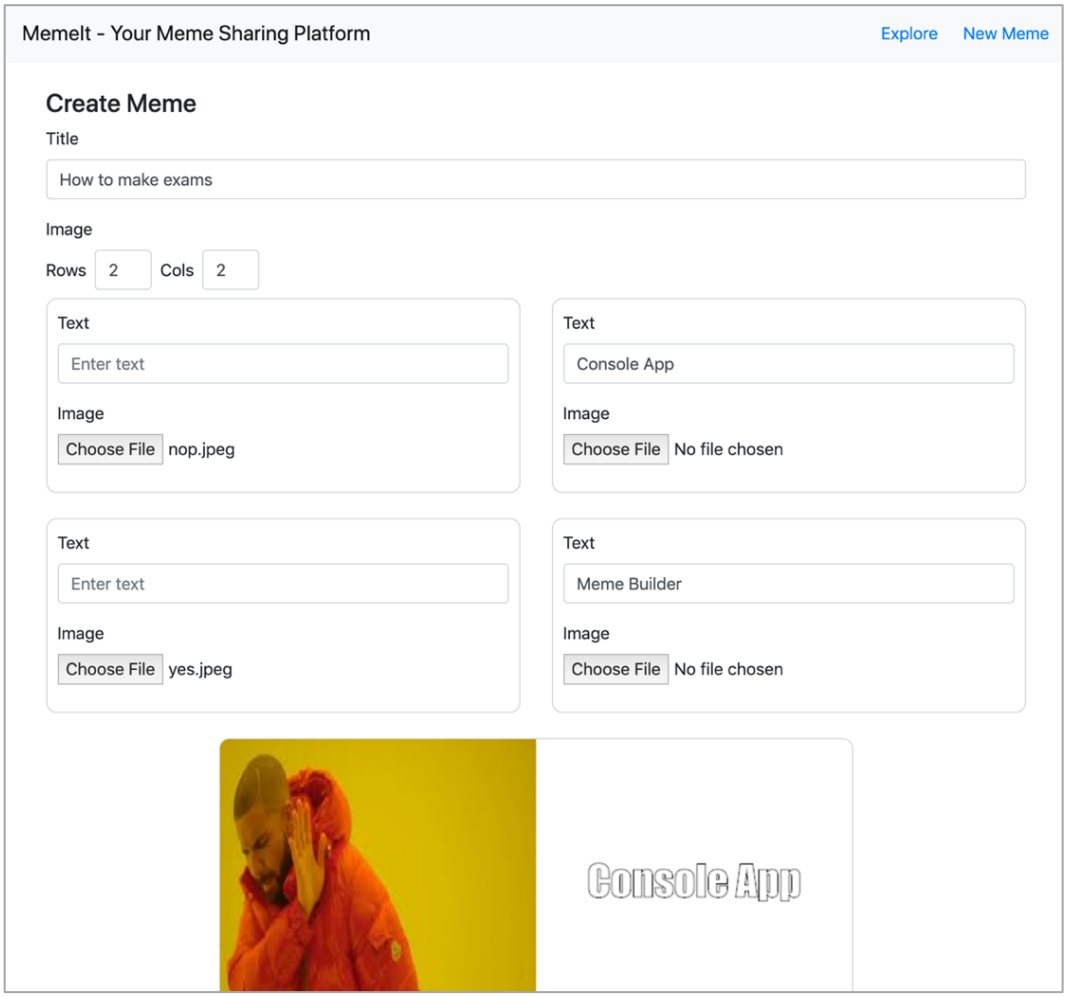
The user should be able to upload an image and add a text for each box.
A meme must have a title and at least one box. A box should have either a text or background.

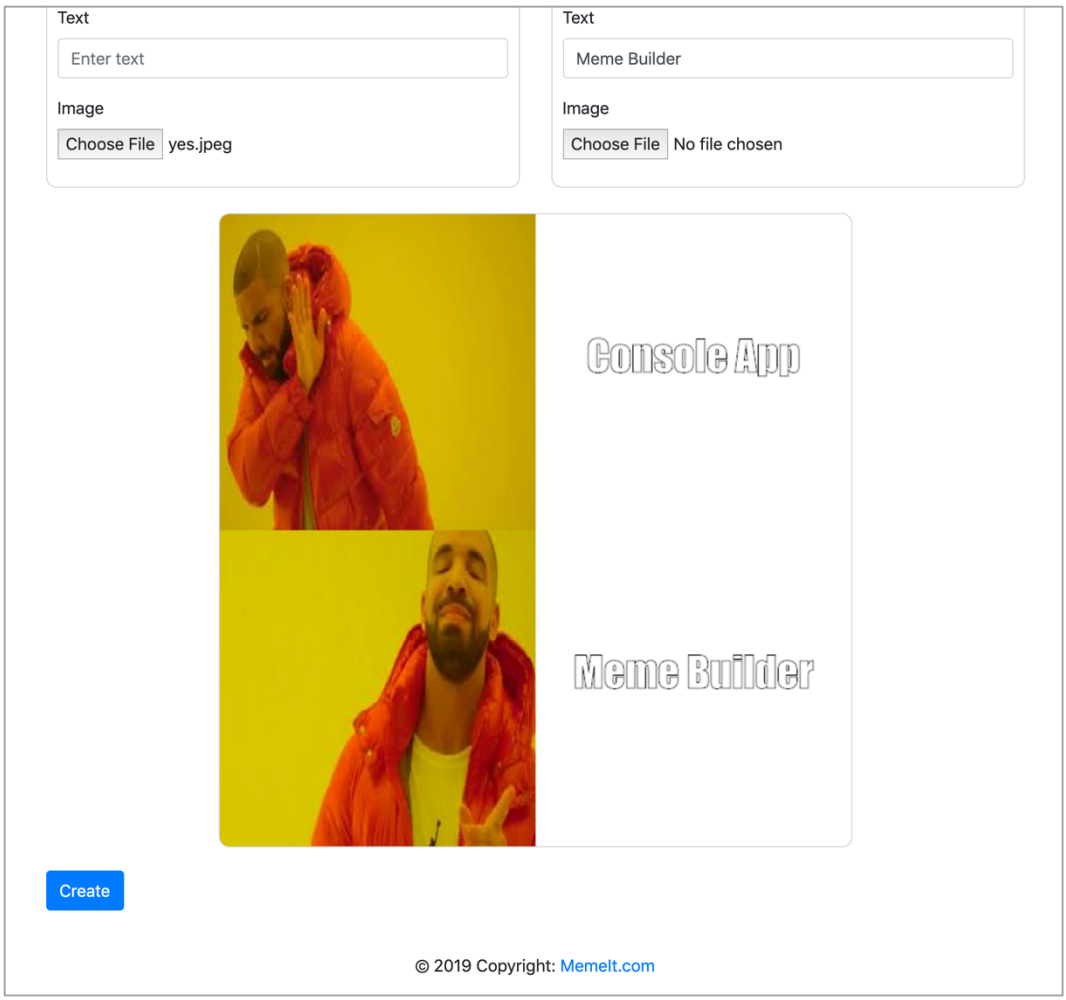


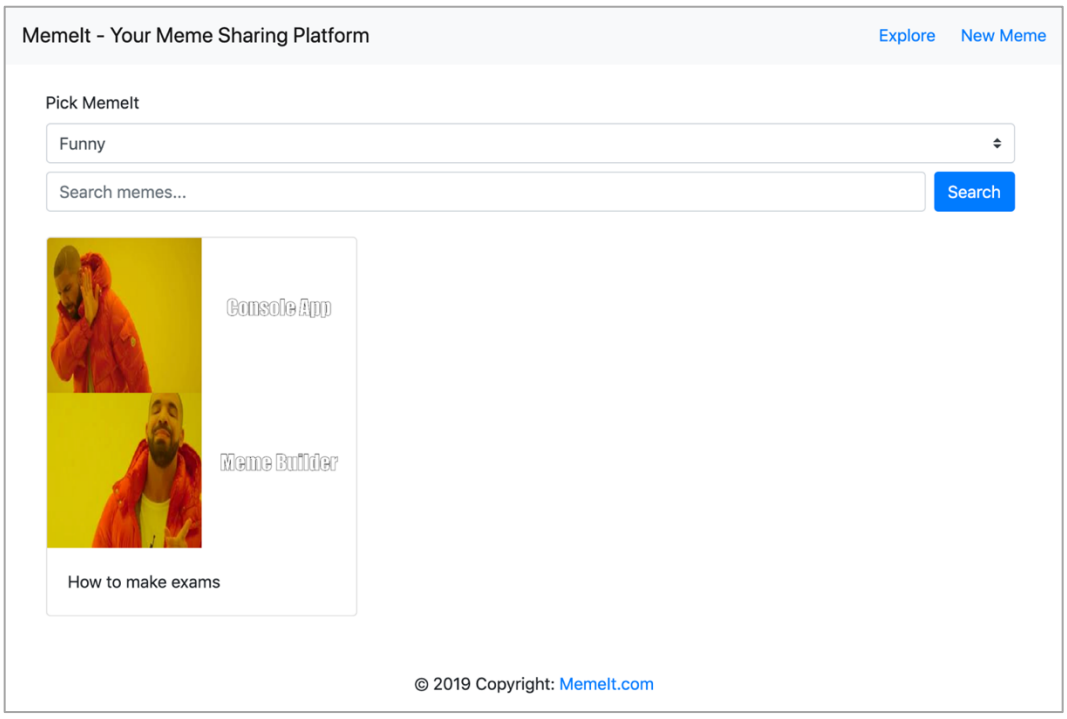
Hints:

1. Remove the file input field and put two new number fields for rows and cols
2. For each row and col (you will use nested **ngFor*'s) display a text and file inputs
3. *Bind* whenever they *change* and display all available images and texts in a canvas
4. Convert the canvas to file (*blob*) and use it as an image to pass the backend

Usage:




Text
Enter text
Image
Choose File
yes.jpeg
Text
Meme Builder
Image
Choose File
No file chosen
Console App
Meme Builder
Create
© 2019 Copyright: MemeIt.com


MemeIt - Your Meme Sharing Platform
Explore
New Meme
Pick MemeIt
Funny
Search memes...
Search
Console App
Meme Builder
How to make exams
© 2019 Copyright: MemeIt.com