# esta función obtiene 2 listados: uno de las imágenes de la API y otro de favoritos, ambos en formato Card, y los dibuja en el template 'home.html'.

def home(request):

    lista\_cards=services.getAllImages()

    images = []

    favourite\_list = []

    for card in lista\_cards:

        images.append(card)

    return render(request, 'home.html', { 'images': images, 'favourite\_list': favourite\_list })

en el archivo views.py se modificó la función home y agregamos la variable “lista\_cards “que tiene un listado de imágenes en formato card usando la función “getAllImages()”, que esta en el archivo “services.py”

recorrimos esta lista para obtener cada elemento de la misma y lo agregamos a la lista “images = []” que estaba vacía.

# función que devuelve un listado de cards. Cada card representa una imagen de la API de Pokemon

def getAllImages():

    # debe ejecutar los siguientes pasos:

    lista\_imagenes\_crudas=transport.getAllImages()  # 1) traer un listado de imágenes crudas desde la API (ver transport.py)

    lista\_cards=[]

    for dato in lista\_imagenes\_crudas:    # 2) convertir cada img. en una card.

        card=translator.fromRequestIntoCard(dato)

        lista\_cards.append(card)

    # 3) añadirlas a un nuevo listado que, finalmente, se retornará con todas las card encontradas.

    return lista\_cards

en el archivo services.py se modifico la función getAllImages(),y y le agregamos lista\_imagenes\_crudas , que tiene un listado de imaenes crudas de la API usando la función getAllImages(),que esta en el archivotransport.py.

creamos lista\_cards=[],para guardar todas las cards y devolverla en la función.

recorrimos esta lista para obtener cada elemento de la misma, convertimos cada imagen obtenida en una card con la función fromRequestIntoCard del archivo translator.py y lo agregamos a la lista “lista\_cards=[]” que estaba vacía.

  card=translator.fromRequestIntoCard(dato)

        tipos=[]

        tipos\_cartas= card.types

        for tipo in tipos\_cartas:

          tipos.append(get\_type\_icon\_url\_by\_name(tipo))

        card.types\_imgs=tipos

        lista\_cards.append(card)

se agregaron los tipos de pokemón, agregamos una lista vacía para guardar el tipo de pokemón, ya que no lo tenía antes guardado.

div class="card mb-3 ms-6 {% if 'fire' in img.types %}border-danger{% elif 'grass' in img.types %}border-success{% elif 'water' in img.types %}border-primary{% else %}border-warning{% endif %} " style="max-width: 600px; ">

a partir de la información anterior se pudieron filtrar las imágenes con el condicional, por el tipo de pokemón y así también se le asignó el color correspondiente a cada card.

def filterByType(type\_filter):

    filtered\_cards = []

    lista\_imagenes=getAllImages()

    type\_filter=type\_filter.lower()

    for card in lista\_imagenes:

        if type\_filter in card.types:# debe verificar si la casa de la card coincide con la recibida por parámetro. Si es así, se añade al listado de filtered\_cards.

            filtered\_cards.append(card)

    return filtered\_cards

        return redirect('home')

en la función filterByType agregamos la variable lista\_imagenes que trae todas las cards de los pokemón, la recorrimos para clasificarlos por su tipo. filtered\_cards es la variable en la cual se guardaron los pókemon ya clasificados por su tipo

if type != '':

        images = services.filterByType(type) # debe traer un listado filtrado de imágenes, segun si es o contiene ese tipo.

        favourite\_list = []

def filterByCharacter(name):

    filtered\_cards = []

    name=name.lower()

    for card in getAllImages():

        if name in card.name:  # debe verificar si el name está contenido en el nombre de la card, antes de agregarlo al listado de filtered\_cards.

            filtered\_cards.append(card)

    return filtered\_cards

en la función filterByCharacter usamos la variable name en la cual asignamos el nombre del pokemon que buscamos

 if name != '':

        images = services.filterByCharacter(name)

        favourite\_list = []