```python
#APOD Viewer
import tkinter, requests, webbrowser
from tkinter import filedialog
from tkcalendar import DateEntry
from PIL import ImageTk, Image
from io import BytesIO

#Define window
root = tkinter.Tk()
root.title('APOD Photo Viewer')
root.iconbitmap('rocket.ico')

#Define fonts and colors
text_font = ('Times New Roman', 14)
nasa_blue = "#043c93"
nasa_light_blue = "#7aa5d3"
nasa_red = "#ff1923"
nasa_white = "#ffffff"
root.config(bg=nasa_blue)

#Define functions
def get_request():
    """Get request data from NASA APOD API"""
    global response

    #Set the parameters for the request
    url = 'https://api.nasa.gov/planetary/apod'
    api_key = 'DEMO_KEY' #USE YOUR OWN API KEY!!!!
    date = calander.get_date()
    querystring = {'api_key':api_key, 'date':date}

    #Call the request and turn it into a python usable format
    response = requests.request("GET", url, params=querystring)
    response = response.json()

    #Update output labels
    set_info()


def set_info():
    """Update output labels based on API call"""
    #Example response
    '''{'copyright': 'Robert Gendler', 'date': '2020-08-04', 'explanation': "Distorted galaxy NGC 2442 can be found in the southern constellation of the flying fish,
    (Piscis) Volans. Located about 50 million light-years away, the galaxy's two spiral arms extending from a pronounced central bar have a hook-like appearance in
    wide-field images. But this mosaicked close-up, constructed from Hubble Space Telescope and European Southern Observatory data, follows the galaxy's structure in
    amazing detail. Obscuring dust lanes, young blue star clusters and reddish star forming regions surround a core of yellowish light from an older population of
    stars. The sharp image data also reveal more distant background galaxies seen right through NGC 2442's star clusters and nebulae. The image spans about 75,000
    light-years at the estimated distance of NGC 2442.", 'hdurl': 'https://apod.nasa.gov/apod/image/2008/NGC2442_HstGendler_2400.jpg', 'media_type': 'image',
    'service_version': 'v1', 'title': 'NGC 2442: Galaxy in Volans', 'url': 'https://apod.nasa.gov/apod/image/2008/NGC2442_HstGendler_960.jpg'}'''

    #Update the picture date and explanation
    picture_date.config(text=response['date'])
    picture_explanation.config(text=response['explanation'])

    #We need to use 3 images in other functions; an img, a thumb, and a full_img
    global img
    global thumb
    global full_img
    url = response['url']

    if response['media_type'] == 'image':
        #Grab the photo that is stored in our response.
        img_response = requests.get(url, stream=True)

        #Get the content of the response and use BytesIO to open it as an an image
        #Kepp a reference to this img as this is what we can use to save the image
        (Image not PhotoImage)
        #Create the full screen image for a second window
        img_data = img_response.content
        img = Image.open(BytesIO(img_data))

        full_img = ImageTk.PhotoImage(img)

        #Create the thumbnail for the main screen
        thumb_data = img_response.content
        thumb = Image.open(BytesIO(thumb_data))
        thumb.thumbnail((200,200))
        thumb = ImageTk.PhotoImage(thumb)

        #Set the thumbnail image
        picture_label.config(image=thumb)
    elif response['media_type'] == 'video':
        picture_label.config(text=url, image='')
        webbrowser.open(url)


def full_photo():
    """Open the full size photo in a new window"""
    top = tkinter.Toplevel()
    top.title('Full APOD Photo')
    top.iconbitmap('rocket.ico')

    #Load the full image to the top window
    img_label = tkinter.Label(top, image=full_img)
    img_label.pack()


def save_photo():
    """Save the desired photo"""
    save_name = filedialog.asksaveasfilename(initialdir="./", title="Save Image",
    filetypes=(("JPEG", "*.jpg"), ("All Files", "*.*")))
    img.save(save_name + ".jpg")


#Define layout
#Create frames
input_frame = tkinter.Frame(root, bg=nasa_blue)
output_frame = tkinter.Frame(root, bg=nasa_white)
input_frame.pack()
output_frame.pack(padx=50, pady=(0,25))

#Layout for the input frame
calander = DateEntry(input_frame, width=10, font=text_font, background=nasa_blue,
foreground=nasa_white)
submit_button = tkinter.Button(input_frame, text="Submit", font=text_font,
bg=nasa_light_blue, command=get_request)
full_button = tkinter.Button(input_frame, text="Full Photo", font=text_font,
bg=nasa_light_blue, command=full_photo)
save_button = tkinter.Button(input_frame, text="Save Photo", font=text_font,
bg=nasa_light_blue, command=save_photo)
quit_button = tkinter.Button(input_frame, text="Exit", font=text_font, bg=nasa_red,
command=root.destroy)

calander.grid(row=0, column=0, padx=5, pady=10)
submit_button.grid(row=0, column=1, padx=5, pady=10, ipadx=35)
```

```
full_button.grid(row=0, column=2, padx=5, pady=10, ipadx=25)
save_button.grid(row=0, column=3, padx=5, pady=10, ipadx=25)
quit_button.grid(row=0, column=4, padx=5, pady=10, ipadx=50)

#Layout for the output frame
picture_date = tkinter.Label(output_frame, font=text_font, bg=nasa_white)
picture_explanation = tkinter.Label(output_frame, font=text_font, bg=nasa_white, wraplength=600)
picture_label = tkinter.Label(output_frame)

picture_date.grid(row=1, column=1, padx=10)
picture_explanation.grid(row=0, column=0, padx=10, pady=10, rowspan=2)
picture_label.grid(row=0, column=1, padx=10, pady=10)

#Get today's photo to start with
get_request()

#Run the root window's main loop
root.mainloop()
```