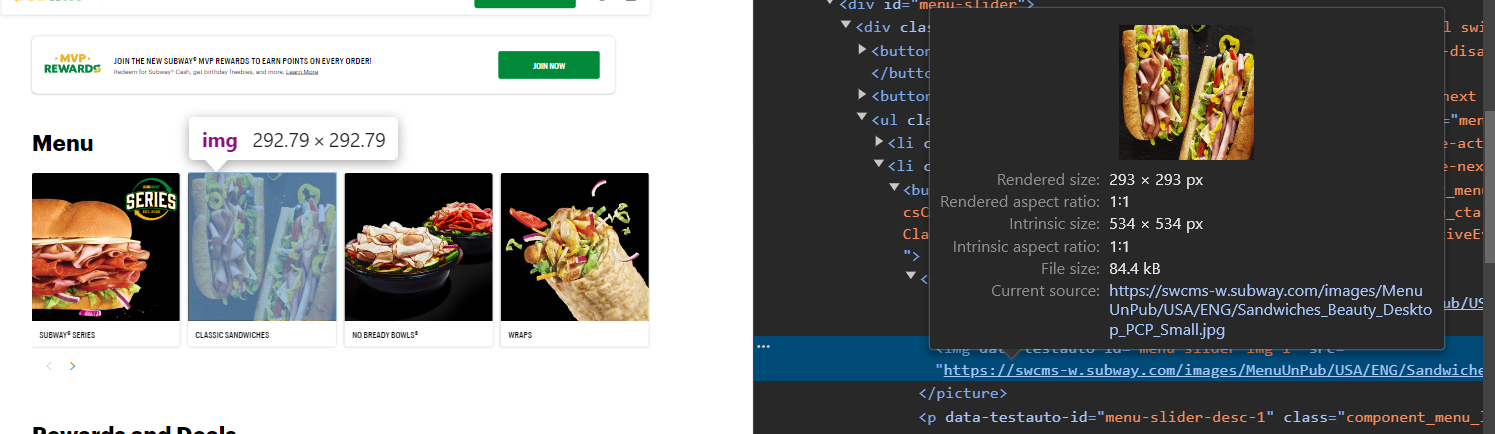
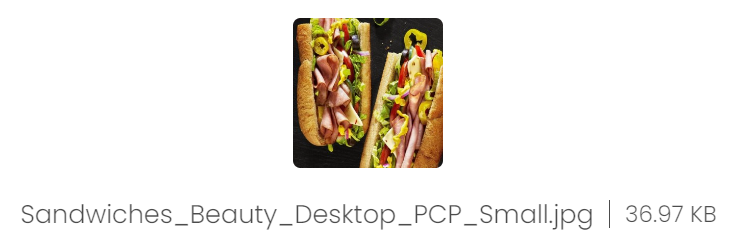
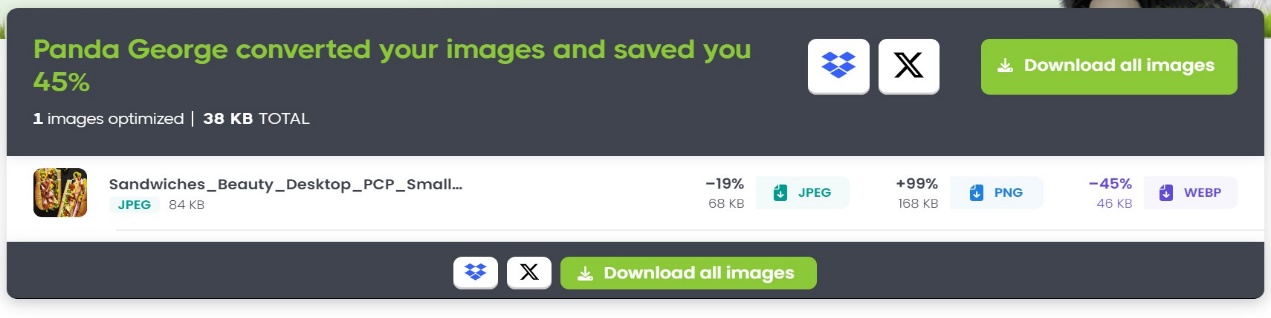
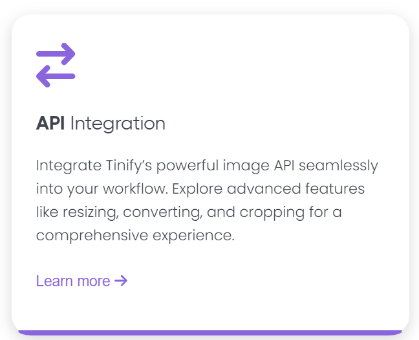
**REPORT ON IMAGE OPTIMIZATION: -**

1. **IMAGE SIZE / DIMENSION** are not coming as per the dimension required. Hence leading to Bigger image size (KB) and leading to greater load time.  
   EX:-  
     
   ****  
     
   We can see here that the required Image size is **293x293 px,** that is the biggest it can go on desktop and greater screen resolution devices. But the dimension we are getting is of **534x534 px**, which is of **84.4 kb  
     
   Solution:-  
   i)** Resize the image to the maximum required container size ( preferable w.r.t desktop screen).  
   Here, I tried to resize the above image to 293x293 and below is the output:-   
      
   🡪 **almost 50% reduction in data size** of the image  
   which ultimately leads to **faster load time**  and **rendering**

**ii) Compress** the image to its maximum capacity without degrading the quality.

* ****Use the **WEBP** format, since as per below conversion we can see that the **compressibility point reached to almost 50%** of actual image ie. **( 46kb earlier 84kb )** rather than going for PNG or JPG ( also suggested by Lighthouse).  
    
  
* Use of Online compression tools like Tiny png (<https://tinypng.com/)> which also provide API service (with pricing )  
  

But, since we are getting the Image URL in the API response, it doesn’t make sense to implement at the code project level, hence it must be optimized   
( both resized & compressed) from the API end point.