

Image Resizer

Overview

Image Resizer Tool enables a person to resize an image in three simple steps.

1. Upload an image
2. Enter valid dimensions for the new image.
3. Click on resize

You can also download the resized image using Download button.

Functional Requirements

1. Create a web interface that accepts and resizes images.
2. Input canvas should display the uploaded image.
3. Output canvas should display the resized image
4. Resize operation should be done using a backend server in python.
5. User can download the resized image.

Non-Functional Requirements

1. User can only enter valid inputs for the dimensions.
2. Latency of the API call should be small.
3. Styling needs to be applied to view a very large image.
4. Download is only enabled when the resized image is generated.

Out Of Scope

The current application can be improved with the following suggestions which could not be implemented because of time constraints.

1. The User Interface can be made more responsive to accommodate all screen sizes.
2. If a user enters only one dimension, the other dimension can be calculated automatically to maintain the initial image aspect ratios.
3. Error can be thrown from the endpoint in case of failures, which can be handled in UI to show different dialogue boxes.

Architecture Overview

Backend

The Backend endpoint is created using **Flask**. Flask is a micro web framework written in **Python**. It is classified as a microframework because it does not require particular tools or libraries. The functional logic for the API is written using the **OpenCV** library. The Functions receive new image dimensions i.e., height and width, and image encoded using **base64** from the frontend using a **POST API call**. The base 64 encoded string is converted to an image, resized, and then encoded back to the base64 string that is sent in response structure to the frontend.

```
* Serving Flask app "__main__" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: on

* Restarting with stat
* Running on http://localhost:9000/ (Press CTRL+C to quit)
127.0.0.1 - - [12/Feb/2023 16:46:14] "POST /resizer HTTP/1.1" 200 -
```

Frontend

The web service is created using **React**, **JavaScript**, and **CSS** for styling. Various functions, functional components, styles, hooks and tags are used in order to build the web application.


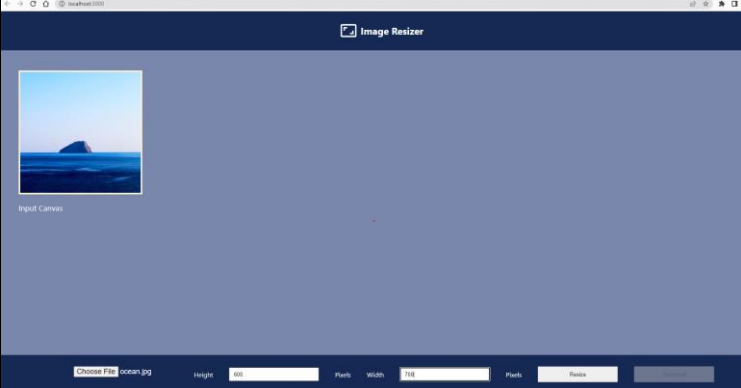
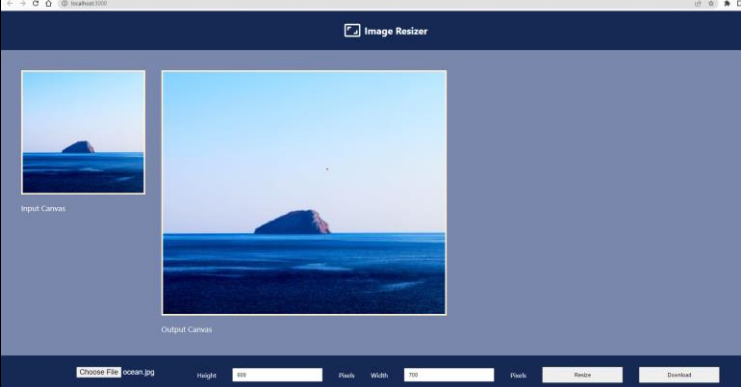
The complete webpage is divided mainly into three sections.

- **Header** – this displays the tool name and icon and is fixed to the top of the page.
- **Center Body** – this is the main section where initially a welcome text is displayed to show the user steps to resize. It displays the input canvas i.e. the uploaded image and the resized image based on the user inputs. This section of the page can be turned into a scrollable section when the resized/uploaded image is very big.
- **Footer**- this section of the page controls all the functionality of the web application and is always fixed to the bottom of the viewing window. It has an upload button used by the user to upload the image, and fields to accept user input for height and width. The resized button and download button, which are disabled in the beginning are also present in this section only. The resized button is enabled once the user enters valid inputs for height and width, and the download button is enabled once the resized image is generated. The resized image is downloaded as "Resized.jfif".

The uploaded image is encoded to base64 string and is passed to backend along with height and width inputs when the user clicks on the Resize button. After a successful API call, the output canvas displays the resized image which is received as a base64 Encoded string.

Start the Application

- You need to host the backend of the application by starting the Jupyter notebook.
- Start the frontend application using npm start. It runs on the local host port 3000.

Stages of the Application	Web Application view	Comments
Initial View		This is the initial view of the application. The Center body displays a welcome message along with the steps to resize an image. The Resize and Download buttons are disabled initially. The user needs to click on the upload button to start the process.
Image is uploaded		The welcome text disappears and the uploaded image is displayed as an input canvas. Once the user enters valid inputs, resize button is enabled. The user needs to click on the resized button to generate the resized image.
Resized Image Generated		The output Canvas displays the resized image. Once resized image is generated, the download button is enabled and the user can download the resized image.
A recorded demo demonstration the working of the tool is attached here .		