Comp2Dust Project Overview

**1.Description**

Project **Comp2Dust**

Prepared by Yiğit Şık

Contact: [yigit6958@gmail.com](mailto:yigit6958@gmail.com)

Git: <https://github.com/YigitSIK/Comp2Dust>

**2.Elevator Pitch**

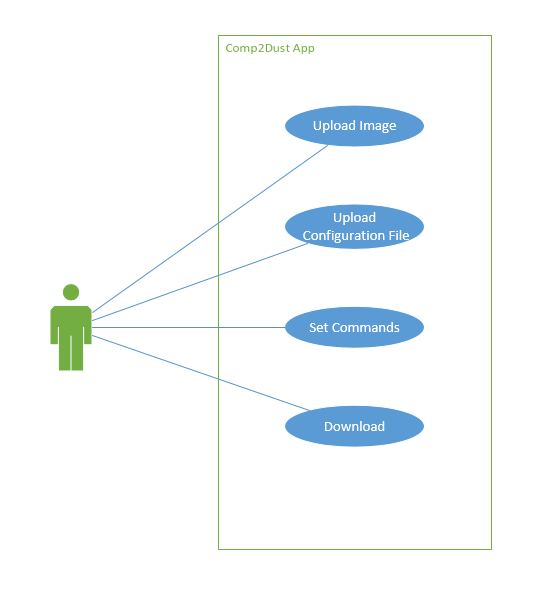
Comp2Dust online image optimizer uses various image compression engines to optimize and modify JPEG, PNG and GIF files while keeping the required level of quality.

**3.Overview**

JPEG, GIF and PNG formats make most of the entire internet’s image traffic. That is why this web site will be aiming these three format. Due to increasing upload numbers each day, it become a need to optimize content.

Comp2Dust web site is an online image compressor available for every user who has an internet connection and any popular browser such as Firefox, Mozilla, Chrome ... Our intended users are people who need smaller sizes for their environment. And since we have covered three main formats that today’s internet consist of, we have common users.

**Use case of the application:**



**Upload Image:** User uploads up to 100 Jpeg, Png and Gif files

**Upload Configuration File:** User uploads a configuration file, such as a custom quantization table

**Set Commands:** User sets options before compression

**Download:** User downloads processed images

**4.Requirements**

* MongoDB database with Atlas cloud service to keep image data but not images themselves.
* In terms of user interface, HTML, CSS, and Bootstrap library will be used to illustrate the system attractively.
* Server side compression function depends on npm package called “compress-images”.
* Another npm package to called “body-parser” to get parsed information from HTML to server side.
* Axios library will be used for sending http requests.
* Multer library for uploading multi-part form data.
* Zip-local npm package to zip folders
* On the server side, express js, and its middlewares, will be used to handle routes and mongoose to connect database.
* Site will be hosted on Heroku servers.

The System’s core functionality is provided by npm package “compress-images”.

The package provides several compression engines as APIs.

Comp2Dust image compressor uses fallowing engines due to their consistency:

JPG Engines:

* mozjpeg
* jpegtran

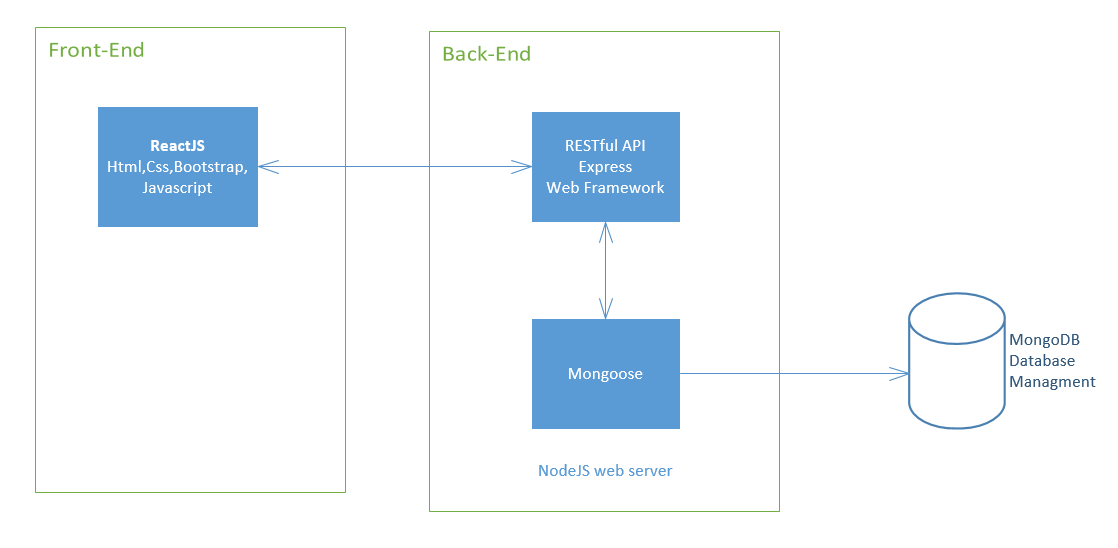
PNG Engines:

* pngquant

GIF Engines:

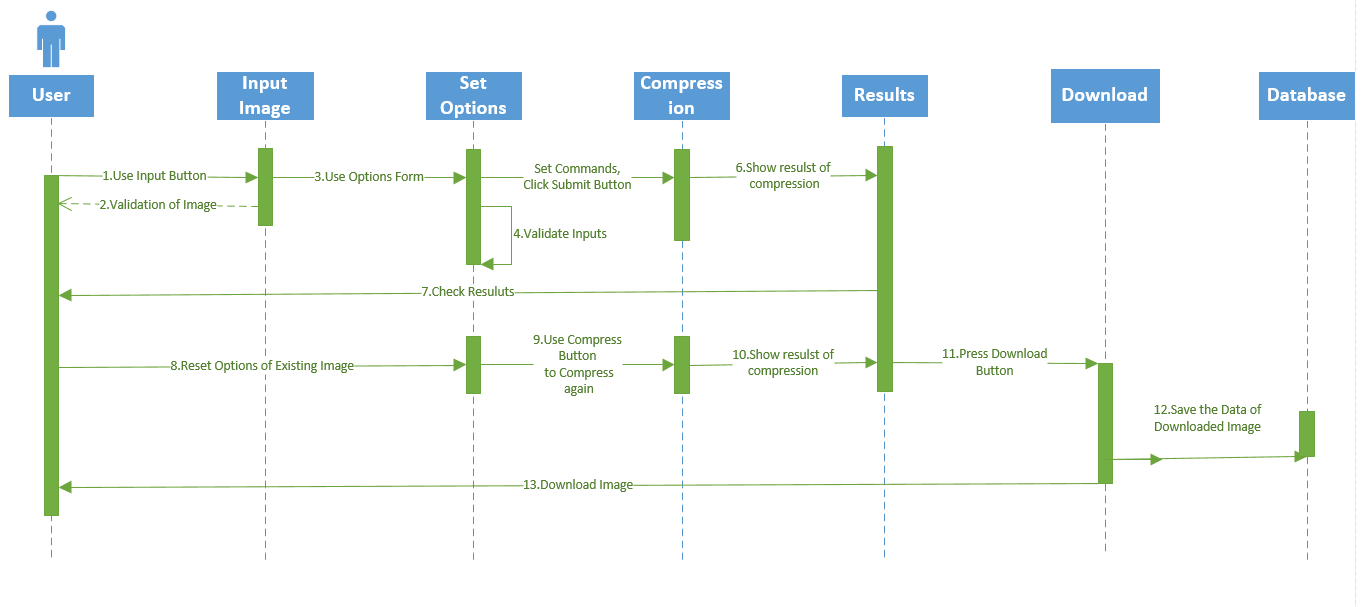
* gifsicle

**Block Diagram of the system is given below**



**5.Functionality**

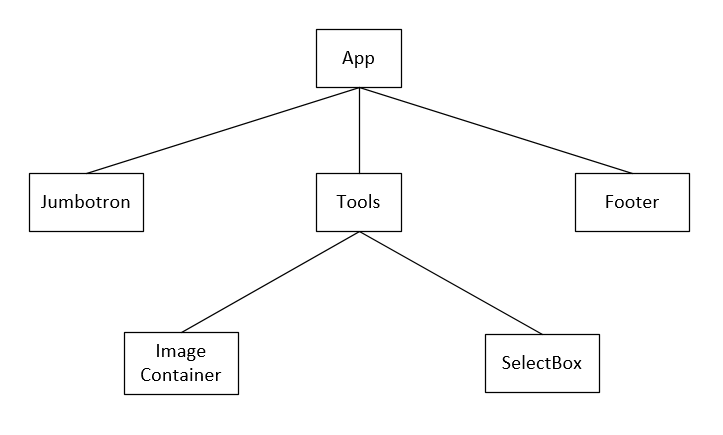
Here is an example to general flow of the application



**6.Design**

Comp2Dust is a single page web application that has been designed with ReactJS. Thus it is highly modifiable and good looking.

Components tree of Comp2Dust is given below:



**App:** Covers all content

**Jumbotron:** Contains Header part of the application

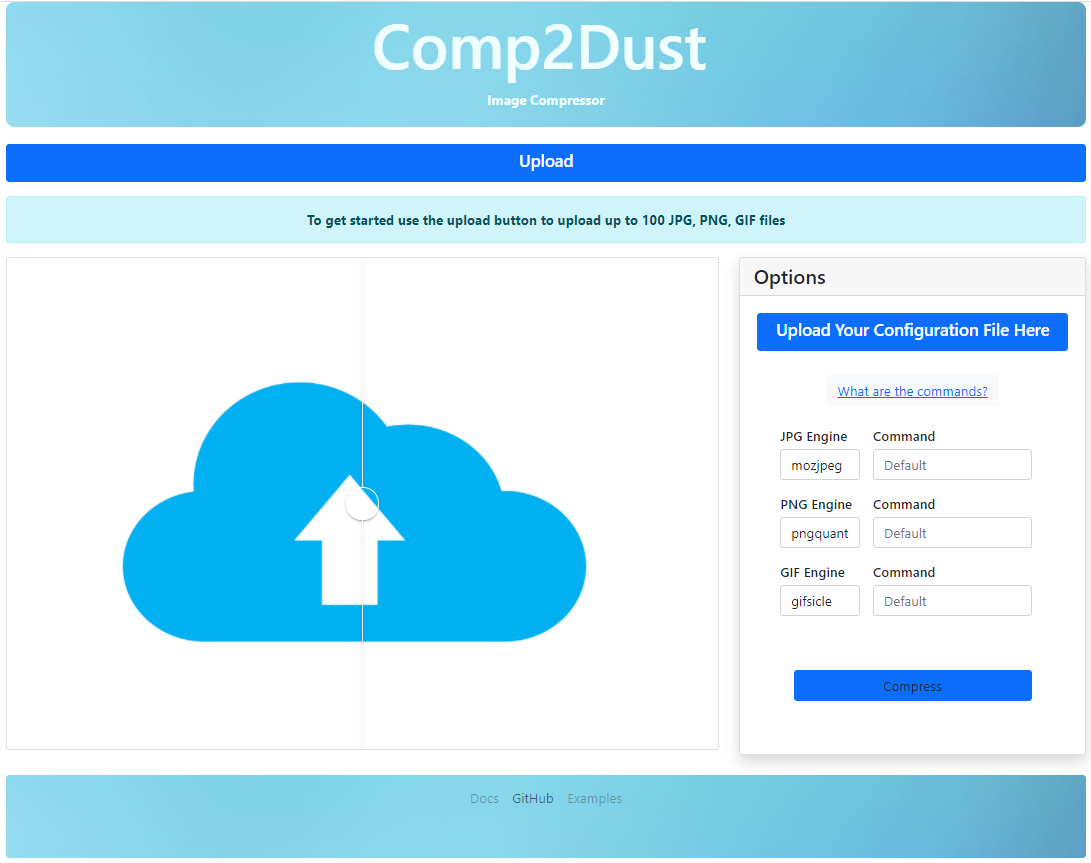
**Tools:** This is where magic happens, contains all the image displayers and options

**Footer:** This is the bottom part of the application, contains nav-bar links

**Image Container:** The Main image displayer with comparison feature is here as an image gallery.

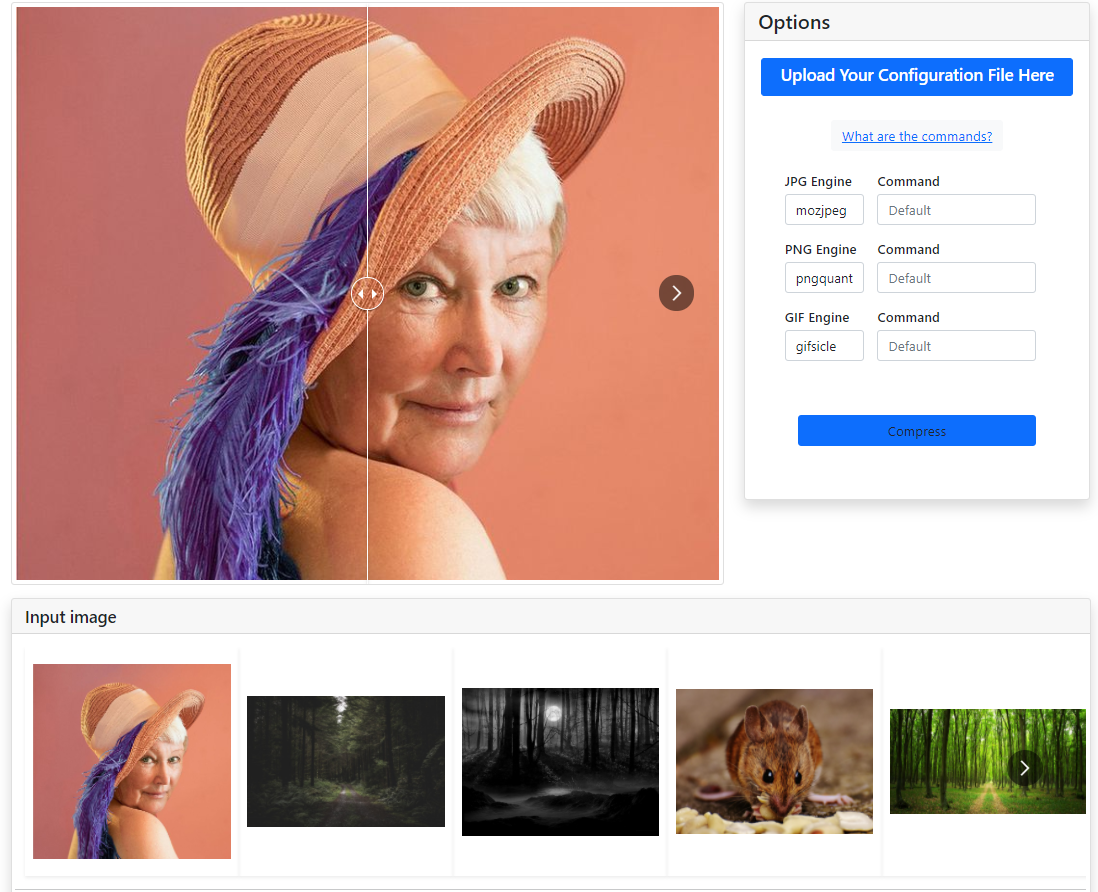
**SelectBox:** At this part of the application user enters commands, chooses compression engines.

At first glance to app, there is a pretty landing page like the one below

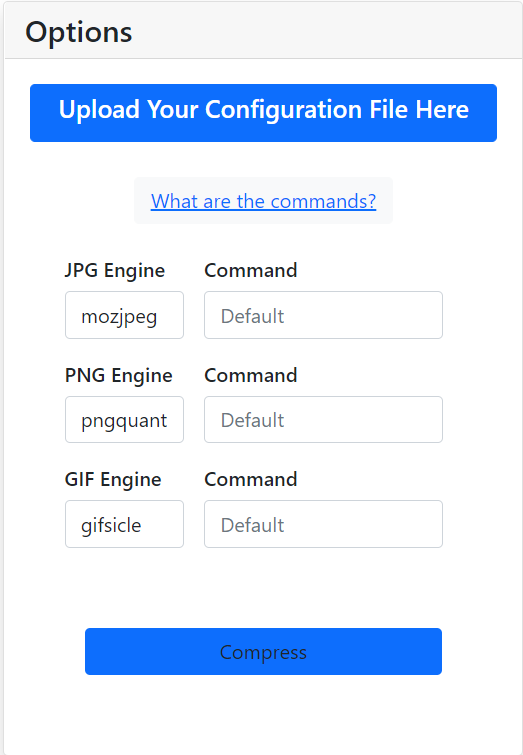


As the Info Box says, user starts with uploading his images through the input box.

Right after user see upload progress through progress bar which will show up below. On images loaded, another carousel displays those images

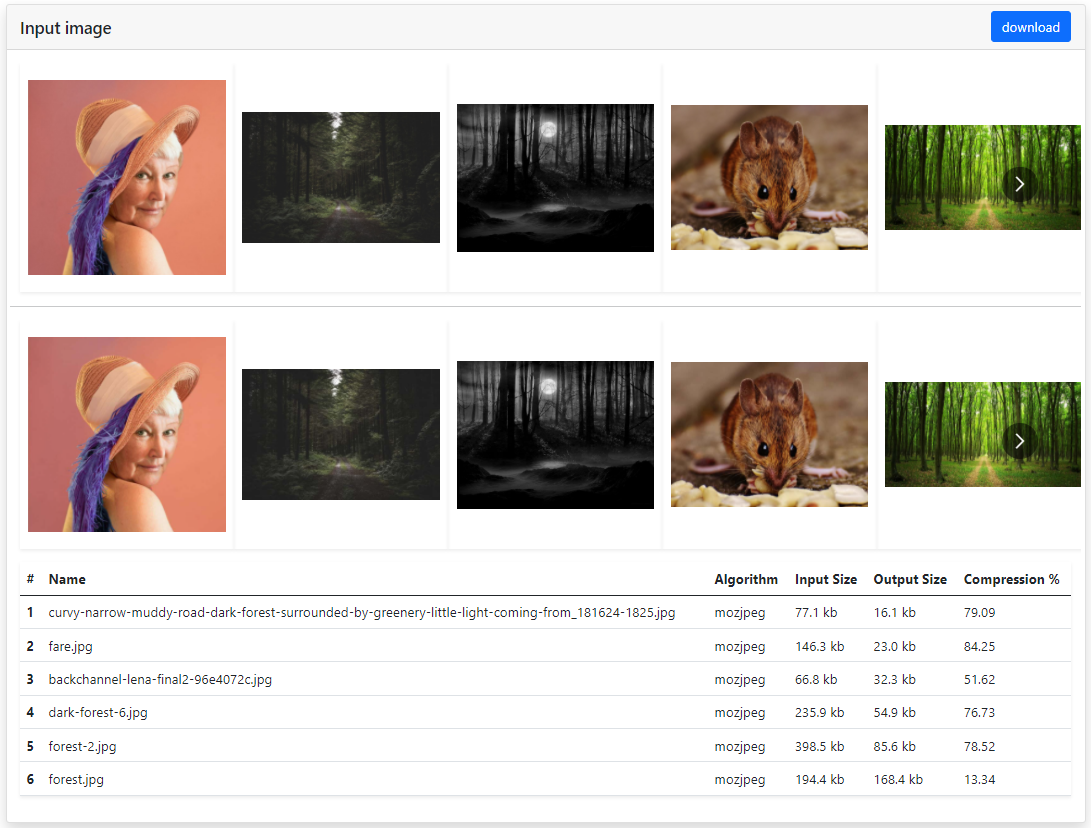


Application provides a graphical user interface that takes user inputs, transferring them to compression API.



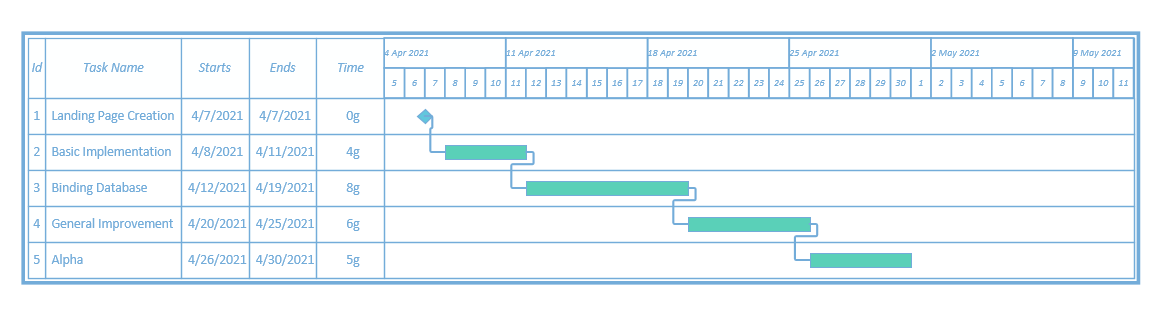
* Commands of the application will be discussed in user manual

After compression, results are displayed with statistics and ready to download.



**7.Milestones**

**Alpha**



* As a first task there will be the basic design of constructed landing page.
* By the end of 2nd task application should perform the main function crudely
* At third step working algorithm should be able to send data to MongoDB database.
* At fourt step, any occurence of bugs should be fixed, input validations and exception handling task should be implemented.
* In Alpha section application should be properly working and its user interface should be revised for presentation.

**Final**

