Team 12 Project 3: Colorization Prototype

[Project Info]
| Language: Python
| Topic: B/W image colorization

[Person-Hour Estimation]
| Total: 15 Hours
| This is a rough estimate time from choosing a good topic for the Project 3. Since topic was not given to the team, we suspect quite of time will be spent choosing a topic. Total 15 hours include discussion, meeting, coding, documentation, and research needed for Project 3. About 5 person hours to decide on the topic and research about the planned topic. About 10 person-hours to discuss the plans and building a prototype.

[Person-Hour Log]
| General communication among team members: 60 Minutes
| Topic Discussion: 30 Minutes
| Topic Reserach: 60 Minutes
| RGB Library Research: 30 Minutes
| Python Project Setup: 60 Minutes
| Github Repo Setup: 60 Minutes
| Coding: 15 Hours

| Total: 20 Hours

[Code Documentation]
| This project is using the source code of @richzhang (Github) as a base code.
| Project is written in Python
| We will be adding UI for the program and implementing functions for UI.
| This project will be a prototype of [Project 4]
| This program converts B/W image to a RGB image (colorization). Users are able to choose a file and colorize the image. Image will be resized to fit the canvas for the program. Program "communicates" with the neural network and receives corresponding data into a variable to colorize the image and resizes to its original dimension of the image. Converts to an RGB image based on the ab data for the image display. Finally, program displays the final RGB image to the user.

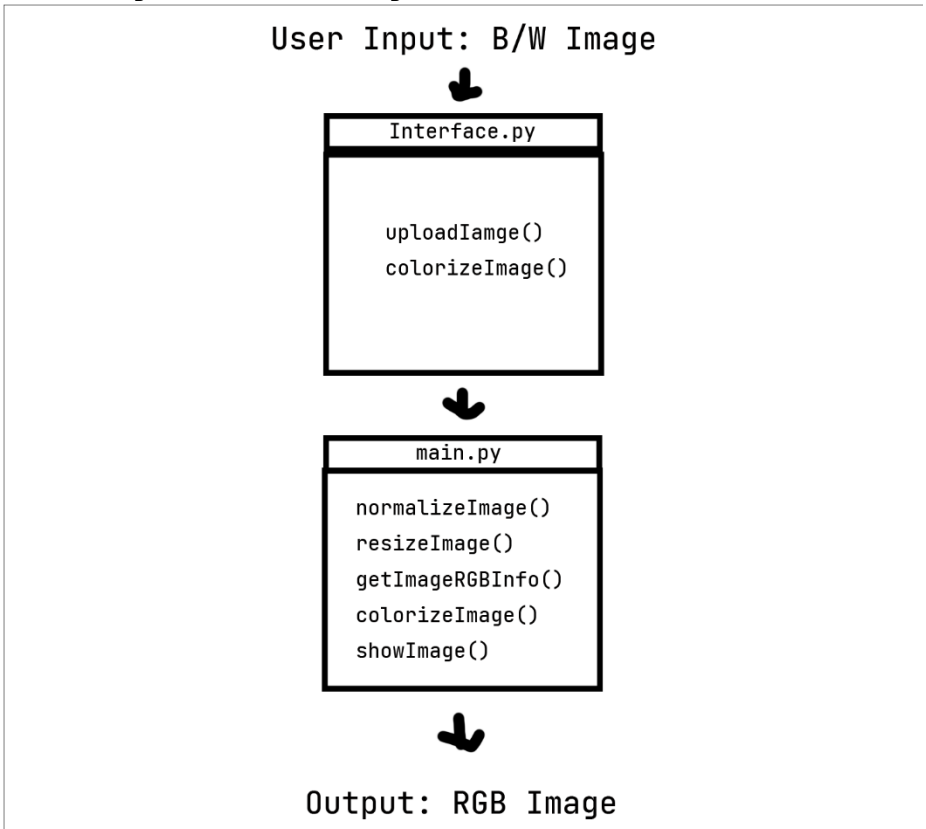
| main.py is the default process handler for colorization.

| interface.py is the default UI/Form handler for user to view.

[Design Paradigm]
| The design paradigm for this project is ‘Data-Driven’. We tried to come up with unique idea that could help the individual team members to learn and improve throughout project 3 and project 4. For this, we have decided to use python which we have not used for previous projects and use the “colorization” source to create a program to colorize the black and white images to RGB images. We knew that there are many resources (or libraries) to use, and we have found the source through Github that had contained the digitalized data of the RGB images. (https://github.com/richzhang/colorization/tree/caffe/models).
| We thought object-oriented program was done often during class and projects, so this technology was a great opportunity for us to focus more in to [Data-Driven] design paradigm since this technology is using the previous data which is the black and white image and processing the data with the algorithm by using the library found on Github; then, it will produce the colorized image of the original black and white image.
| Our team has decided to work on the user interface of the program and possibly add a feature of converting black and white videos to RGB videos. However, we will mainly focus on the technology of processing the data (black and white images) and producing the colorized image and displaying to the user. So, this will be a great chance to get an experience of [Data-Driven] paradigm project as well as use of libraries. Additionally, we will also be able to focus on the user interface of the program since it is not a typical huge object-oriented programming project.

[Software Architecture]
| The software architecture for our project 3 is “Layered Architecture”. The reason for choosing [Layered Architecture] is that our project 3 will be a program that converts the black and white image to an RGB image (colorization) which has a clear input and output of the program. The input will be the black and white image which is the original image the user is trying to colorize, and the output will be the RGB/colorized image of the original image after processing the algorithm to the image. This program will not have any process where the process done is different by input.
| Our program will have a same algorithm that is applied to the image, but the input will be very different. User’s choice of image and applying the algorithm (process) which is colorization, will be a great fit as a [Layered Architecture]. The application will also interact with only a few of the modules and libraries. All of the images (data) will be processed with the same algorithm.
| Another great reason for the choice of [Layered Architecture] is that we will be having user interface for the program to have the user control the program. As mentioned, user will be able to choose a file (black and white image) to convert it (colorize) to an RGB image. This process is same throughout any black and white image input, and it will go through the equal process (layers), so we believe [Layered Architecture] is a great fit for our project 3. Finally, for handling any errors and for modifications, [Layered Architecture] software architecture will allow us to quickly develop and make changes when needed.

[UML Diagram - Class Diagram]



[Design Pattern]

| For our team's design pattern, we will be using 'structural design pattern' for developing our application, and it will specifically be 'decorator Pattern'. Reason for choosing [decorator pattern] is that we will be developing an application that will convert (or colorize) the black and white image of the user's choice to a colorized (RGB) image. This application will have a set algorithm for using the digitalized RGB data of the image and colorize the image, so this process will not change. However, we will be adding custom features and user interface to have user more control of the application rather than the application just giving the RGB image (output).

| The initial classes, for our team's application would the be main.py where it colorizes the black and white image, will remain unchanged since it is a set algorithm for the application. However, the [decorator] will be modified and will be added to the application for user interface and custom features such as colorizing the black and white video instead of an image.

| Our application, especially the prototype (project 3) is depending on the technology and the data of colorization of the image, so there will not be much of "object" related alterations. Our interface.py class or any other addition will be acting as the wrapper of the main.py (our main algorithm processing class). We will be focusing on interface and additional use of the library we are using and add-on to the application (prototype) of our final project.