SHILPA'S PROJECT CONTRIBUTIONS

NU ID: 001406044

FILTER/DOWNLOAD STAGE:

- Organized the filter/download stage as a borderpane with the following:
    - The top and bottom nodes contain the header and footnote content,
    - The left node contains the *Apply Filter* dropdown as a menu button,
    - The right node contains the *Download As* dropdown as a menu button,
    - The center node contains a scrollpane for displaying images of any size, scrollable as necessary.
- These items are present within the *processImage* method of the *ImageConversions* class.

FILTER FUNCTIONALITY:

- The filter functionality is implemented using JMagick's *MagickImage* class.
- Within the *Apply filter* dropdown, the following menu items are added for the user to select from: GrayScale, ImplodeScale, LogColorScale, CharcoalScale, and OriginalScale.
- Each filter request is handled by processing the original image and displaying the output on the scrollpane. In this manner, the user can easily see the effect of the filter in real-time. The feed for the scrollpane is obtained by creating a temporary *FilterTemp* directory (if not exists), implemented via the *createtempdir* method in the *ImgUtils* class.
- **GrayScale filter**: Applied a gray scale filter on the original image. Used the *MagickImage* class and set *GRAYColorspace* in the *setColorspace* method of the *QuantizeInfo* class. This functionality is implemented within the overridden *filter* method of the *GrayScale* class.
- **ImplodeScale filter**: Imploded the original image from the center. Used *implodeImage* method of the *MagickImage* class with a value of -0.5, the negative value is used to explode the image. This functionality is implemented within the overridden *filter* method of the *ImplodeScale* class.
- **LogColorScale filter**: Applied a Logcolor scale filter on the original image. Used the *MagickImage* class and set the *LogColorspace* int value using the *setColorspace* method of the *QuantizeInfo* class. This functionality is implemented within the overridden *filter* method of the *LogColorScale* class.
- **CharcoalScale filter**: Implemented the charcoal filter that highlights the edges in the image to result in a charcoal-drawing effect. Used the *charcoalImage* method of the *MagickImage* class with a pixel neighborhood radius of 2 and standard deviation of 1. This functionality is implemented within the overridden *filter* method of the *CharcoalScale* class.
- **OriginalScale**: This filter is used to reset the effect of any filter applied and reproduces the original image in the scrollpane. This functionality is implemented within the overridden *filter* method of the *OriginalScale* class.
- The implementation allows the user to go back and forth between filters of a given image and decide which ones to download.
- When the user clicks one of the filters, the click triggers the corresponding filtering above

DOWNLOAD FUNCTIONALITY:

- The download functionality is implemented using the *writeImage* and *setFileName* methods of the *MagickImage* class. The argument to the *setFileName* accepts the file extension from the dropdown, and the file *tag* from the *Tags* class.
- Within the *Download As* dropdown, the following menu items are added for the user to specify the download format: JPG, PNG, BMP, and GIF.
- Each download request converts the image into the requested format and stores the resulting image with filter tags (*_grayfiltered_, _implode_, _logfiltered_, _charcoal_, _original_*) added to the filename as applicable.
- All download requests from the user are served in the *Downloaded_Images* directory (if not already present), via the *createdir* method in the *ImgUtils* class.
- When the user clicks one of the filters, the click triggers the corresponding download using the *commonFormat()* method of the format class.

DISPLAY TEXT:

- Created labels in the filter/download stage with appropriate styles for displaying heading and footnotes. The heading and footnotes were pushed into HBoxes and centered in the top and bottom respectively. These are present in the *processImage* method of the *ImageConversions* class.
- Displayed project title label in green within the application homepage. This is present within the *start* method of the inherited *JavaFXProj* class.

IMAGE UTILITIES:

- The *ImgUtils* class encapsulates the class variables which can be set by the methods in the class. It provides various utilities such as *createdir(), createtempdir(), deleteDirectory(File dir)* that support the filter/download functionality of the application.
- *ImgUtils -> createdir()*: This method checks for the presence of a download directory relative to the uploaded directory and creates one if it doesn't already exist.
- *ImgUtils -> createtempdir()*: This method checks for the presence of a filter temporary directory relative to the uploaded directory and creates one if it doesn't already exist. This is used to load filtered images into the scrollpane and is pointed to by the file pointer in the *Tags* class at various stages of the application.
- *ImgUtils -> deleteDirectory(File dir)*: This method deletes the files inside the directory along with the directory. It is used only to delete the temporary directory used for storing filter images during transition. This method is triggered upon closing the Filter/Download stage.
- *Tags* class: This class is used to tag the filter state of the images (*_grayfiltered_, _implode_, _logfiltered_, _charcoal_, _original_*) to let the user identify the filter tag associated with the downloaded images. In addition, the class also maintains a SelectedFile to update the state of the file pointer after every filter is applied.

DOCUMENTATION/TESTING/COMMENTING/SOFTWARE:

- The documentation, test cases, and comments for the filter/download/ImgUtils functionality were completed and integrated into the overall project documentation, test case document, and code comments respectively.
- Successfully installed and configured ImageMagick/JMagick software after several weeks of effort debugging installation issues.