Methodology for the project

1. Masking.py
   - I read all the groundtruth values from the csv files and stored it in dataframes for x_coordinate and x_radi respectively.
   - I have used the ground truth values to mask the images. Using a simple formula of subtracting the x_coordinate and x_radi to find the left corner of the bounding box and then adding x_coordinate and x_radi to find the right corner of the bounding box.
   - I used the same technique for both Iris and Pupil bounding boxes. Once the bounding boxes were defined, I used the ImageDraw module to draw ellipses around the boxes.

2. Zipper.py
   - Used a simple function to zip everything together so that uploading becomes easier.

3. Ipynb file
   a. Setting up the images:
      - I set up all paths for my image data and resized the image to 256x256 for easier computation. Too low a resolution would result in loss of features and too high a resolution might result in Overfitting.
      - While setting up the images, I resized it, converted the Background to RGB and then normalized pixel values. I used the transpose of the dimensions C,W,H where C being 3 channeled RGB values and fed it to the PyTorch tensor.

   b. Setting up the hyper-parameters:
      - Split the data set into train test and validation with a ratio of 0.3
      - Used 4 num workers (Using excess causes the program to freeze on colab)
      - Used a batch size of 10, wanted to try using Colab TPU but was too slow so went with a lower batch size.
      - Initially started with 100 epochs
      - Learning rate of 0.01. I find this the best threshold when training a U-Net model

   c. Dice Loss function:
      - Used Dice Loss to find the overlap between ground truth and predicted masks.
      - I used this because I felt that Pupil is a rare class and the Dataset was small so Dice loss would suit as a better metric.

   d. I have used the U-Net architecture for my model since this model was specifically designed for this test case. The down-sampling coupled with the up-sampling works really well when dealing with images especially when segmenting them. It is also useful when dealing with small features in images especially pupils which are generally small.

e. I have used ADAM optimizer because it works well with high dimensional data(images), and it is adept at handling noisy gradients effectively when updating the learning rate.

f. A scheduler was used that adjusts the Learning Rate when it plateaus after some epochs.

g. Training the model:
    - I stopped training the model after 50 epochs. I did this because I noticed that my training loss was consistently reducing but not the validation loss. If I kept on training the model further, it would have resulted in overfitting which is not ideal.

h. I created a visualization of the results, the first row being the original image, second being the ground-truth mask and third being the predicted mask.

i. Evaluation of results:
    - I used Canny Edge detection to extract the edges of the masks
    - Used Hough Circle transform to the output of the edge detection to detect circular shapes.
    - If any circles are found, the diameter is estimated by multiplying the radius of the circle by 2
    - If no circles are found, a default value of 80 is added.
    - I used this technique for both Ground-Truth masks and Predicted masks for Iris and Pupil respectively.

j. I then calculate the Mean Absolute Error in gt and prediction by using MAE formula. This gives me the Iris Diameter error and Pupil Diameter error.

k. Finally I calculated the IoU by doing 1-Dice_Loss_Val.