Steps I followed:

1) Preparing dataset (images, annotated labels) for YOLOv5x

we wanted to make entire images corresponding with their classes bounding box description inside all the frame in terms of text file with same name.

- Test file generation : First we making free text files for each images in the dataset.

  Filling class and bounding box desctription to the text files:-
- Reading given json file that described annotated labels.
- Found 4 keys in the json data dictionary (taking only keys: ‘images’ and ‘annotations’). In annotations key, all given represented bounding box descriptios are in the form of COCO format’
- Using python code, we are filling all text files (attaching data_prep.py file) (here, COCO format bbox description, we have to covert into YOLO format – attaching pictures that i drawn and convertion equation for better understanding,C2Y1.jpg, C2Y2.jpg)

2) Downloading yolov5 repository from github

3) Making a file ‘dataset’ and making subfolder ‘images’ and ‘labels’. Again move inside each subfolder and making two subfolders named by ‘train’ and ‘val’. Fill images and corresponding text files (make sure that your validation data limitage based on entire data observations numbers)

4) Download pre-trained yolov5x sized by ~200 Mb which contains learnt weights on coco dataset(yolov5x.pt)

5) Creating a dataset.yaml file to read number of classes& to read train and validation images (attaching dataset.yaml in repository)

6)Training :

- Installing all requirements files inside the yolov5 directory(attaching requirements.txt file in repository)
- Running the training the python file named by train.py (attaching in repository).using ArgumentParser function, give inputs train the model:
  Inputs such as, image size, batch size, pretrained weights, dataset, epochs, device for using multiple gpus, configuration file, optimizer-adam etc...
- During the training, creating a runs file indicates all training descriptions like data visualization, validation accuracy metrics, last optimized weights, best weights file and results maP and other metrics etc(attacching first six epochs descriptioned results.csv in repository)

7) Testing :

Attaching detect.py file to detect for images and videos by giving ArgumentParser function for sources (images or video), image size, confident threshold(0.45 default), iou threshold(0.25 default) etc.

Attaching inputs : images to detect(img1.jpg, img2.jpg,img3.jpg), and video to detect(video.mp4)
Attaching outputs : detect1.jpg, detect2.jpg, detect3.jpg, detect_video.mp4