

JUIT-OLX

Project report submitted in partial fulfilment of the requirement for
the degree of Bachelor of Technology

in

Computer Science and Engineering/Information Technology

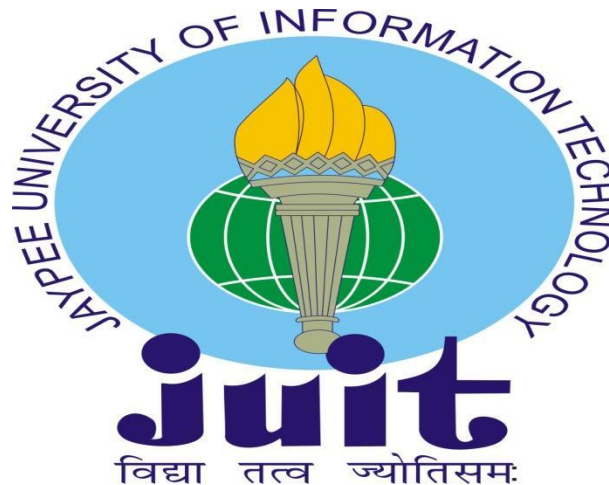
By

Mayank Ekaghara 191406
Aditya Tomar 191431

Under the supervision of

Dr. Monika Bharti

to



Department of Computer Science & Engineering and Information
Technology

**Jaypee University of Information Technology Waknaghat,
Solan-173234, Himachal Pradesh**

CERTIFICATE

This is to certify that the work which is being presented in the project report titled “**JUIT-OLX**” in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science And Engineering and submitted to the Department of Computer Science And Engineering, Jaypee University of Information Technology, Waknaghat is an authentic record of work carried out by “Aditya Tomar, 191431” and “Mayank Ekaghara, 191406” during the period from January 2023 to May 2023 under the supervision of Dr. Monika Bharti, Department of Computer Science and Engineering, Jaypee University of Information Technology, Waknaghat.

Aditya Tomar

(191431)

Mayank Ekaghara

(191406)

The above statement made is correct to the best of my knowledge.

Dr. Monika Bharti

Assistant Professor (SG)

Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date:

Type of Document (Tick): PhD Thesis M.Tech Dissertation/ Report B.Tech Project Report Paper

Name: _____ Department: _____ Enrolment No _____

Contact No. _____ E-mail. _____

Name of the Supervisor: _____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): _____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor)

Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	<ul style="list-style-type: none"> • All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String 		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature

Librarian

.....

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

ACKNOWLEDGEMENT

Firstly, I express our heartiest thanks and gratefulness to almighty God for his divine blessing makes it possible to complete the project work successfully.

I am really grateful and wish our profound indebtedness to Supervisor **Dr. Monika Bharti, Assistant Professor (SG)**, Department of CSE/IT Jaypee University of Information Technology, Wakhnaghat. Deep Knowledge & keen interest of our supervisor in the field of “**MERN Stack**” & “**Machine Learning**” to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project. I would like to express my heartiest gratitude to **Dr. Monika Bharti**, Department of CSE/IT, for her kind help to finish my project.

I would also generously welcome each one of those individuals who has helped me straightforwardly or in a roundabout way in making this project a win. In this unique situation, I also want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking. Finally, I must acknowledge with due respect the constant support and patience of our parents.

Aditya Tomar
(191431)

Mayank Ekaghara
(191406)

TABLE OF CONTENT

S. No	Title	Page No.
1	Certificate	i
2	Plagiarism Certificate	ii
2	Acknowledgement	iii
3	Table of content	iv
4	List of figures	v
8	abstract	vii
9	Chapter - 1 Introduction	1
10	Chapter - 2 Literature survey	13
11	Chapter - 3 System Design & Development	23
12	Chapter - 4 Experiments & Results Analysis	40
13	Chapter - 5 Conclusions	58
14	References	61

LIST OF FIGURES

S. No.	Figure No.	Description	Page no.
1	1.1	Working of YOLO	6
2	1.2	Prediction Result	6
3	1.3	Comparison of various models	7
4	1.4	Comparison of backbones. Accuracy, billions of operations	8
5	1.5	Object Detection to recognize animals with YOLO in an agriculture application with Viso Suite	9
6	1.6	YOLOv3 prediction results with values above 0.1 being too large and the detection accuracy is too low.	12
7	3.1	Software Development and Life Cycle	23
8	3.2	Gantt chart	24
9	3.3	Tech stack used	27
10	3.4	Database design	34
11	3.5	Website flowchart or action diagram	35
12	3.6	The Architecture of YOLO	39
13	4.1	Node Download page	40
14	4.2	Starting React	41
15	4.3	Login page	43
16	4.4	Sell page	44
17	4.5	Cart page	45

18	4.6	Home page	46
19	4.7	Category page	47
20	4.8	Profile page	48
21	4.9	Checkout page	49
22	4.10	Item recognition code	50
23	4.11	Input Image 1	52
24	4.12	Output image 1	53
25	4.13	Item Recognised 1	53
26	4.14	Input image 2	54
27	4.15	Output image 2	54
28	4.16	Recognised image 2	55
29	4.17	Evaluation of several detectors	56
30	4.18	COCO dataset performance	56

ABSTRACT

juitOLX is an e-commerce website designed to provide students with a sustainable and cost-effective way to buy and sell items they need. To promote sustainability and legal compliance, the website has incorporated a machine learning (ML) model using the YOLOv3 algorithm. This model detects illegal objects in images uploaded by students before they are listed on the website. juitOLX is built using the MERN stack, which includes MongoDB, ExpressJS, ReactJS, and NodeJS, providing a robust and scalable platform with real-time updates for buyers and sellers. The website offers features such as image uploads, search functionality, messaging system, and a seamless and transparent buying and selling experience. By promoting sustainability, responsible consumption, and legal compliance, juitOLX has the potential to revolutionize e-commerce platforms for students.

Chapter-1

INTRODUCTION

1.1 Introduction

The juitOLX is an innovative e-commerce website that aims to facilitate a sustainable and cost-effective way for students to sell and buy items they need. As we all know, many students tend to have unwanted items that they no longer use and these items can often be used by other students who need them. By providing a platform for students to sell their items, juitOLX can help reduce waste and promote a more sustainable lifestyle among students. To ensure that the items being sold on juitOLX are safe and legal, we have incorporated a machine learning (ML) model using the YOLOv3 algorithm. This model is trained to detect illegal objects in images uploaded by students who want to sell their items. The YOLOv3 algorithm is a state-of-the-art object detection algorithm that can identify and locate objects in an image with high accuracy.

The juitOLX platform is built using the MERN stack, which includes MongoDB, ExpressJS, ReactJS, and NodeJS. These technologies provide a robust and scalable platform for the website, with real-time updates to keep buyers and sellers informed of any changes to their transactions. juitOLX provides a wide range of features to make buying and selling items easy and hassle-free for students. The website allows students to upload images and descriptions of the items they want to sell, and interested buyers can browse through these listings and contact the seller directly. The ML model using YOLOv3 ensures that images uploaded by students do not contain any illegal objects before they are listed on the website. To make the buying process easier, juitOLX also provides a search functionality that allows buyers to filter listings based on their preferences. Buyers can search for items based on category, price range, location, and other relevant factors. The website also provides a messaging system that enables buyers and sellers to communicate directly with each other to negotiate terms, arrange for payment and delivery, and resolve any issues that may arise during the transaction. In today's world,

sustainability and responsible consumption are becoming increasingly important. With juitOLX, we aim to promote a more sustainable and responsible lifestyle among students by providing a platform for them to sell and buy items they need. This can help reduce waste and promote the reuse of goods, which in turn can help reduce the carbon footprint of students.

Moreover, the incorporation of a machine learning model using the YOLOv3 algorithm ensures that all items being sold on the platform are safe and legal. This not only protects buyers from purchasing illegal items but also protects sellers from any potential legal issues that may arise. The MERN stack provides a robust and scalable platform for juitOLX, ensuring that it can handle a high volume of traffic and transactions. The real-time updates feature keeps buyers and sellers informed of any changes to their transactions, providing a seamless and transparent buying and selling experience. In conclusion, juitOLX is an innovative platform that has the potential to revolutionize the way students buy and sell items. By promoting sustainability, responsible consumption, and legal compliance, we believe that juitOLX can have a positive impact on students' lives and the environment. We are excited to see how juitOLX develops in the future and hope that it will continue to serve as a model for sustainable and responsible e-commerce platforms.

1.2 Problem Statement

The life of a student is often characterized by limited resources and tight budgets. Students often have items in their rooms that they no longer need or use, including laptops, books, and other electronic devices that are still functional but are of no use to them. These items often end up gathering dust or being thrown away, contributing to wastage and a negative impact on the environment.

At the same time, other students may be in need of these items but may not be able to afford to buy them new. However, the lack of a dedicated platform for students to sell and buy these items makes it challenging for them to connect with potential buyers or sellers. This leads to a lack of awareness and potential buyers or sellers, which further contributes to wastage and a negative impact on the environment.

Moreover, there is also a risk of students attempting to sell illegal items on such platforms, which can lead to legal complications and can put the safety of other students at risk. Therefore, it is essential to have a mechanism in place to ensure that only legal and safe items are sold on such platforms.

In addition to the above, the lack of a platform for students to sell and buy items they need creates a financial burden on students. Students may need items such as laptops or textbooks for their studies but may not be able to afford to buy them new. By providing a platform for students to buy and sell these items, we aim to provide a cost-effective way for students to acquire the items they need, reducing the financial burden on them.

To address these issues, we propose the development of an e-commerce website specifically for students to sell and buy items they need. This website will incorporate a machine learning model using the YOLOv3 algorithm to ensure that only legal and safe items are listed for sale. The website will provide a user-friendly platform for students to list items for sale, search for items to buy, and connect with potential buyers or sellers. By doing so, we hope to promote sustainability, reduce wastage, and facilitate a cost-effective way for students to buy and sell items they need.

1.3 Objectives

- i. Develop an e-commerce website using the MERN stack specifically for students to sell and buy items they need.
- ii. Incorporate a machine learning model using the YOLOv3 algorithm to ensure that all items listed for sale on the platform are legal and safe.
- iii. Provide a user-friendly platform for students to list items for sale, search for items to buy, and connect with potential buyers or sellers.
- iv. Ensure that the website is scalable and can handle a high volume of transactions.
- v. Implement real-time updates to keep buyers and sellers informed of any changes to their transactions.
- vi. Promote sustainability and responsible consumption by encouraging the reuse of goods and reducing wastage.
- vii. Facilitate a cost-effective way for students to buy and sell items they need, reducing financial burdens on students.
- viii. Provide a transparent buying and selling experience, reducing the potential for fraudulent activities.
- ix. Ensure that the website complies with all legal and ethical standards.
- x. Evaluate the success of the website by measuring the number of transactions, user satisfaction, and the impact on sustainability and responsible consumption among students.

1.4 Methodology

A real-time object detection system called YOLOv3 (You Only Look Once, Version 3) recognises particular things in films, live feeds, or still photos. To find an item, the YOLO machine learning system leverages features that a deep convolutional neural network has learned. The YOLO machine learning algorithm has three versions, with the third version being a more accurate version of the first ML method. Versions 1-3 of YOLO were developed by Joseph Redmon and Ali Farhadi. Version 1 of YOLO was made in 2016, while version 3, which is the one that is heavily addressed in this article, was made in 2018. An enhanced version of YOLO and YOLOv2, YOLOv3 is available. The Keras or OpenCV deep learning libraries are used to implement YOLO.

Working of YOLOv3:

YOLO is a Convolutional Neural Network (CNN) that can quickly identify objects. CNNs are classifier-based systems that can analyse incoming images as organized arrays of data and identify relationships between them (see illustration below). The benefit of YOLO is that it is faster than other networks while still maintaining accuracy.

Because it enables the model to view the entire image during testing, its predictions are influenced by the image's overall context. Convolutional neural network methods like YOLO "score" regions according to how closely they resemble predetermined classes. Regions that score highly are reported as positive detections of the class that they most closely match. According on which regions of the video score highly in comparison to predetermined classes of cars, YOLO can be utilised, for instance, in a live traffic feed to identify various types of automobiles.

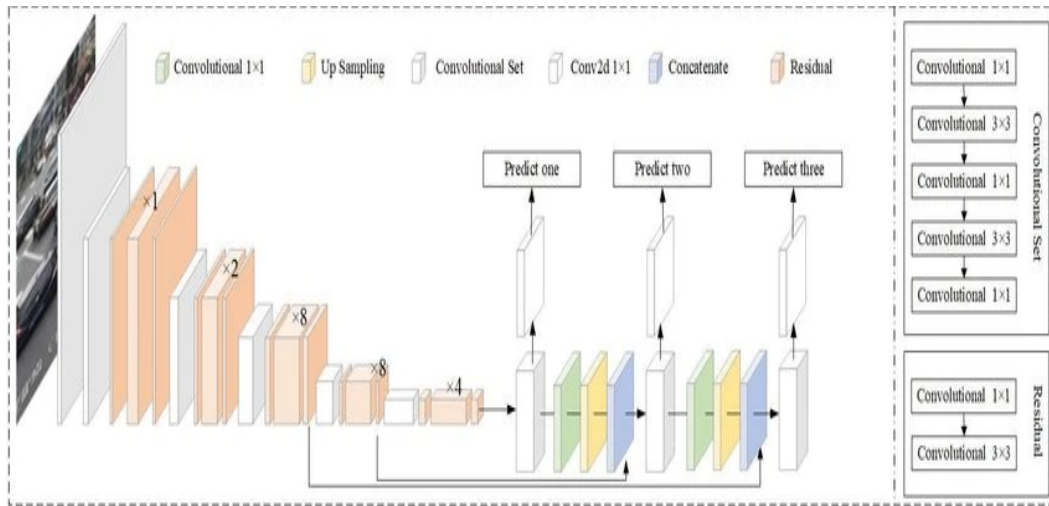


Fig 1.1: Working of YOLO [30]

While YOLOv3 now uses Darknet-53, YOLOv2 was using Darknet-19 as its backbone feature extractor. The YOLO founders Joseph Redmon and Ali Farhadi also created the backbone known as Darknet-53. Darknet-53 is more powerful than Darknet-19 and more effective than rival backbones (ResNet-101 or ResNet-152) since it uses 53 convolutional layers as opposed to the preceding 19 convolutional layers.

Backbone	Top-1	Top-5	Ops	BFLOP/s	FPS
Darknet-19	74.1	91.8	7.29	1246	171
ResNet-101	77.1	93.7	19.7	1039	53
ResNet-152	77.6	93.8	29.4	1090	37
Darknet-53	77.2	93.8	18.7	1457	78

Fig 1.2: Prediction Result [31]

We can see that Darknet-52 is 1.5 times quicker than ResNet101 by referring to the chart presented in the YOLOv3 publication by Redmon and Farhadi. Since it is still as accurate as ResNet-152 but two times faster, the accuracy

shown does not require any trade-off between accuracy and speed between Darknet backbones. In terms of mean average precision (mAP) and intersection over union (IOU) values, YOLOv3 is quick and precise. Compared to other detection techniques with comparable performance, it operates much more quickly.

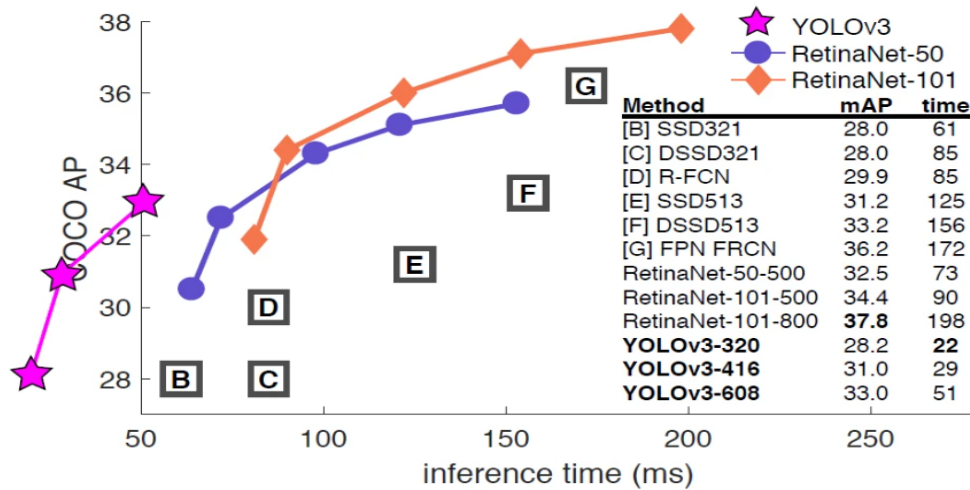


Fig 1.3: Comparison of various models[32]

Precision for Small Objects

The graph that follows, which was updated and obtained from the YOLOv3 publication, displays the average precision (AP) of detecting small, medium, and big images using different algorithms and backbones. For that variable, the higher the AP, the more accurate it is. Because of how bad YOLO was at detecting little items, the precision for small objects in YOLOv2 was unmatched by other algorithms. When compared to other algorithms, such as RetinaNet (21.8) or SSD513, which had the second-lowest AP for small objects, it performed poorly (AP = 5.0).

	backbone	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
<i>Two-stage methods</i>							
Faster R-CNN+++	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI	Inception-ResNet-v2	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	52.1
<i>One-stage methods</i>							
YOLOv2	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5
SSD513	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet	ResNeXt-101-FPN	40.8	61.1	44.1	24.1	44.2	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

Fig 1.4: Comparison of backbone. Accuracy, billions of operations[32]

The AP for small objects increased by 13.3 in YOLOv3, a significant improvement over YOLOv2. Even yet, RetinaNet still outperforms all objects (small, medium, and large) in terms of average precision (AP).

Specificity of Classes

Independent logistic classifiers and binary cross-entropy loss are used by the new YOLOv3 to predict classes during training. These changes enable the use of complicated datasets for YOLOv3 model training, including Microsoft's Open Images Dataset (OID). For photos in the collection, OID has a large number of overlapping labels, including "man" and "person." The multilabel technique used by YOLO v3 enables classes to be more detailed and to have several members for each bounding box. The mathematical function known as a softmax, utilised by YOLOv2, transforms a vector of numbers into a vector of probabilities, with the probability of each value being proportional to the relative scale of each value in the vector.

Each bounding box can only belong to one class when a softmax is used, however this isn't always the case, especially with datasets like OID.

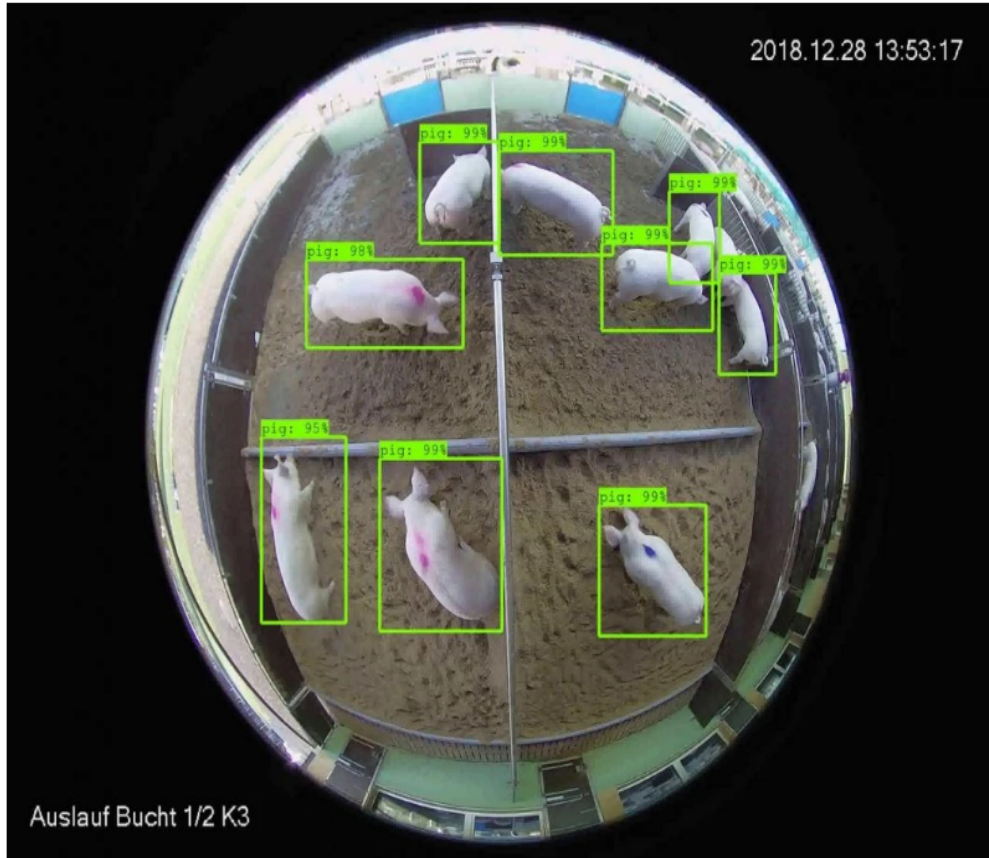


Fig 1.5: Object Detection to recognize animals with YOLO in an agriculture application with Viso Suite

(source:<https://viso.ai/application/animal-monitoring/>)

Disadvantages of YOLOv3 vs. Other Algorithms

Since RetinaNet requires more training time than YOLOv3, the YOLOv3 AP does show a trade-off between speed and accuracy when using YOLO. YOLOv3 is a great alternative for models that can be trained with large datasets because it can be used to detect objects with an accuracy that is comparable to that of RetinaNet when utilising a larger dataset. This can be seen in popular detection models like traffic detection, where a lot of data may be utilised to train the model because there are a lot of photos of various cars. On the other hand, YOLOv3 might not be the best option for using niche models where it can be challenging to find huge datasets.

Making a prediction :

The YOLOv3 architecture's convolutional layers output a detection prediction after feeding the learnt features to a classifier or regressor. The class title, bounding box coordinates, bounding box sizes, and other details are among these properties. The size of the prediction map is exactly the size of the feature map preceding it since the prediction using the YOLO machine-learning method utilises 1 x 1 convolutions (thus the name, "you only look once").

This prediction map is interpreted in YOLOv3 and its other variants so that each cell predicts a specified number of bounding boxes. The cell that will ultimately be in charge of forecasting an item of interest is the one that holds the centre of the ground truth box for that object. The fundamental workings of the prediction architecture are very mathematical.

Anchor Boxes:

Anchor Boxes Although bounding boxes and anchor boxes were briefly mentioned at the beginning of this article, there is more information about how to use and apply them with YOLOv3. Log-space transformations, which are offsets to predetermined "default" bounding boxes, are often predicted by object detectors utilising YOLOv3. These particular bounding boxes are referred to as anchors. To obtain a forecast, the transformations are later applied to the anchor boxes. Particularly, YOLOv3 features three anchors. As a consequence, each cell—which is really referred to as a neuron in more technical terms—predicts three bounding boxes.

Non Maximum suppression:

When more than one bounding box recognises an item as a positive class detection, the object may occasionally be identified more than once. This issue is avoided by non-maximum suppression, which only allows detections if none have already been made. NMS is used to avoid multiple detections by

using the NMS threshold value and confidence threshold value. It is crucial to efficiently use YOLOv3. Here, we provide a quick description of a few of the elements, such as anchor boxes and non-maximum suppression (NMS) values, that enable the predictions. However, this is not an exhaustive list of all the factors that YOLOv3 uses to make a good forecast. Full explanations of YOLOv3's mathematical underpinnings may be found here.

Class Confidence and Box Confidence Scores:

The x , y , w , h , and box confidence score values of each bounding box are present. The confidence score represents the likelihood that a class is included inside that box and the precision of that bounding box. First, the provided image's width and height are used as the bounding box's (w and h) dimensions. Then, all four bounding box values are between 0 and 1, and x and y are offsets of the relevant cell. The YOLOv3 algorithm then applies 20 conditional class probabilities to each cell. The product of the box confidence score and the conditional class probability yields the class confidence score for each final boundary box used as a positive prediction.

The likelihood that the detected item belongs to a specific class (the class being the identification of the object of interest) is known as the conditional class probability in this context. Therefore, the forecast made by YOLOv3 contains three values for h , w , and depth. Then, there is some difficult arithmetic that involves the spatial dimensions of the pictures and the tensors needed to generate boundary box predictions. I recommend reading the YOLOv3 Arxiv paper, which is referenced at the conclusion of this article, if you're curious about what goes on during this phase. The boundary boxes with high confidence ratings (greater than 0.25) are retained as final predictions for the last step.

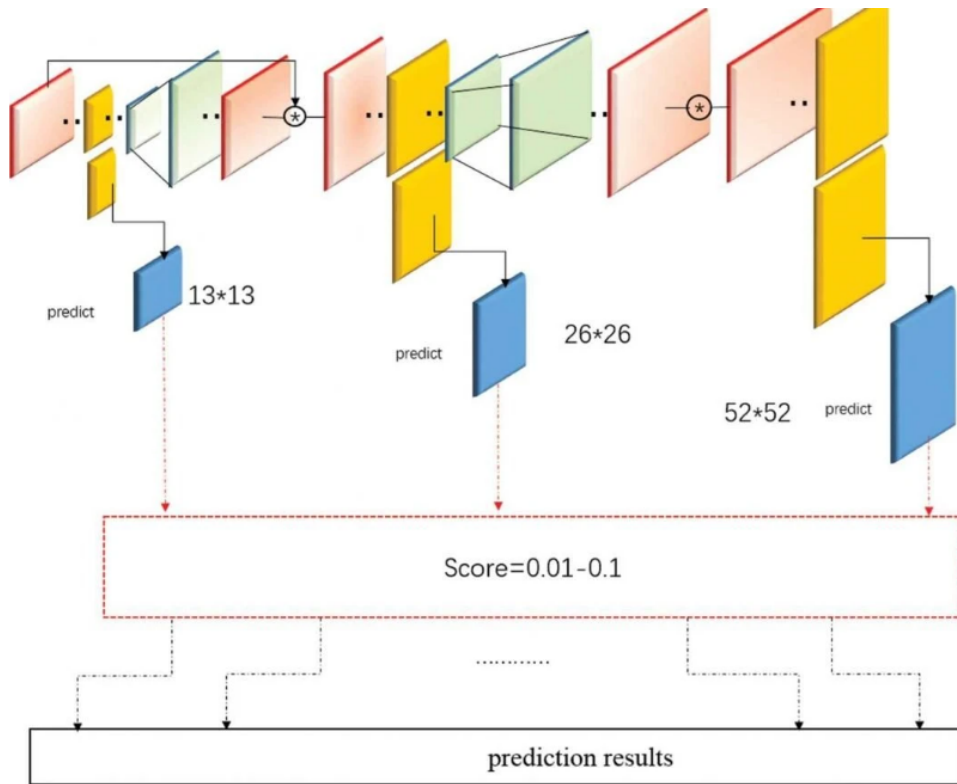


Fig 1.6: YOLOv3 prediction results with values above 0.1 being too large and the detection accuracy is too low.[33]

CHAPTER - 2

LITERATURE SURVEY

INTRODUCTION:

Object detection is a critical task in computer vision and involves identifying and locating objects in images or videos. This task is important for many applications such as autonomous driving, surveillance, robotics and medical imaging. Object detection is a challenging problem because it requires the detection of objects of different sizes, orientations, and scales in complex scenes with occlusions and clutter[1]. In recent years, deep learning algorithms have revolutionised object detection, achieving high-level performance across a wide range of benchmarks.

In this article, we provide an overview of object detection and deep learning algorithms, focusing on the You Only Look Once (YOLO) v3 algorithm[2]. We discuss the main features of the YOLOv3 algorithm, its architecture, and its performance on benchmark datasets[2].

Background information on object detection

Object detection is a computer vision task that involves identifying and locating objects in images or videos. This task can be divided into two subtasks: object classification and object localization. Object classification involves assigning a label to each object in an image or video, while object localization involves finding a bounding box around each object.[1][2][10][15]

Object detection is a challenging problem because it requires the detection of objects of different sizes, orientations, and scales in complex scenes with occlusions and clutter. Traditional object detection methods are based on hand-crafted features and use sliding windows or area suggestions to search for objects. These methods are computationally expensive and require a lot of manual configuration.[5]

In recent years, deep learning algorithms have revolutionised object detection by automatically learning features from data and using extensive training to optimise both object classification and localization tasks. Object detection algorithms based on deep learning can be divided into two categories: two-stage detectors and one-stage detectors.

Two-step detectors, such as region-based convolutional neural networks (R-CNN), Fast R-CNN, and Faster R-CNN, first generate region proposals and then classify and refine them.[6]

YOLO v3 Algorithm:

YOLOv3 stands for You Only Look Once version 3 and is an object recognition algorithm based on deep learning. YOLOv3 is an improvement over YOLOv2, which was an improvement over the original YOLO algorithm. The main improvements of YOLOv3 are better accuracy, better performance on small targets and faster detection.[2][9][11]

Architecture of YOLOv3:

YOLOv3 has a deep neural network architecture consisting of multiple convolutional layers followed by a final fully connected layer. The input to the network is an image that has been passed through convolutional layers, and the output is a set of bounding boxes, each representing an object detected in the image. The YOLOv3 architecture can be divided into three main parts: the core network, the characteristic pyramid network and the detection head.[2]

Backbone Network:

The task of removing features from the input image falls to the backbone network. The Darknet architecture, a modified version of the ResNet architecture, serves as the foundation for the YOLOv3 backbone network. The 53 convolutional layers that make up the Darknet architecture extract information from the input picture at various sizes.[2][9]

Feature Pyramid Network:

The feature pyramid network is in charge of spotting things in the image that have various sizes. The feature pyramid network uses a number of convolutional layers and up-sampling techniques to create a feature pyramid from the output of the backbone network. Each level of the feature pyramid has a distinct amount of spatial resolution and semantic significance.[16]

Detection Head:

The detection head is in charge of identifying items in the picture and forecasting bounding boxes. The detection head applies a series of convolutional layers to the output of the feature pyramid network to produce a collection of detection maps. Each detection map shows the presence of items belonging to a specific class at a specific scale. The further processing of the detection maps results in a collection of bounding boxes that locate the objects in the picture.[2]

YOLOv3 object detection workflow:

The four primary phases of the object identification workflow utilising YOLOv3 are pre-processing, network inference, post-processing, and visualisation.[1][6][2]

Pre-processing:

Pre-processing is the initial stage in the YOLOv3 object detection procedure. The input image is resized to a set size during pre-processing, and the pixel values are normalised. The input picture must have a width and height that are multiples of 32 in order to work with YOLOv3.[24][27]

Network ambiguity:

Network inference is the second phase in the YOLOv3 object detection method. By running the pre-processed picture through the YOLOv3 neural network, a series of detection maps are produced during network inference.[2]

Previous Work on Object Detection using YOLOv3:

The YOLO algorithms, created by Joseph Redmon and his colleagues at the University of Washington, were updated to become YOLOv3. YOLO's initial release, which was meant to be quick and precise, came out in 2015. It did have certain drawbacks, though, namely a propensity for false positives and poor accuracy in identifying tiny objects. Although YOLOv2, the second version, attempted to address some of these shortcomings, there were still some problems with handling occlusions and small object detection.[21]

Redmon and his colleagues unveiled YOLOv3 in 2018, which addressed many of the shortcomings of the earlier iterations. Compared to its predecessors, YOLOv3 is quicker and more precise, and it can identify tiny objects and handle occlusions better. In order to extract features at various sizes, the algorithm also employs a feature pyramid network, which enhances accuracy.[21]

Since the release of YOLOv3, numerous researchers have worked to advance the algorithm and use it in numerous applications. In a 2019 publication titled "Improving YOLOv3 Object Detection with Motion Information," for instance, researchers from Wuhan University[21]

YOLOv3 Application:

Object Recognition in Autonomous Vehicles:

Multiple sensors and cameras are included in self-driving cars so they can sense their surroundings and travel safely. Real-time detection of pedestrians, cars, traffic signs, and traffic lights is possible with YOLOv3. To avoid crashes, decisions like slowing down or halting the automobile can be made using this information. To assist the vehicle with staying in the proper lane, YOLOv3 can also recognise lane lines and road limits.[22]

The requirement for high precision and low latency while employing YOLOv3 in self-driving automobiles is one of the difficulties. Accidents might happen

as a result of any delay or erroneous detection. To fulfil the needs of self-driving automobiles, researchers are thus attempting to enhance the performance of YOLOv3.[3]

Object Recognition in Security Camera Systems:

For security reasons, surveillance systems are employed to keep an eye on public spaces, structures, and private assets. Real-time detection of things including people, cars, and animals is possible with YOLOv3. In the event of questionable activity, this information can be utilised to notify security staff or set off alarms. YOLOv3 has the ability to follow moving objects in the monitoring area.[4]

The capability of YOLOv3 to detect several objects in a single frame is one of its benefits in surveillance systems. As a result, fewer cameras are needed to cover a given area, which lowers the system's cost. However, privacy issues should be taken into account when utilising YOLOv3 and surveillance systems.[12]

Detecting Objects in Medical Imaging:

Various diseases are diagnosed and treated using medical imaging. X-rays, CT scans, and MRI scans are just a few examples of the medical pictures that may be examined for anomalies with YOLOv3. For instance, YOLOv3 is capable of real-time detection of tumours, fractures, and other anomalies. Doctors can utilise this information to help them make precise diagnoses and create effective treatment regimens.[22]

The requirement for excellent accuracy and interpretability while employing YOLOv3 in medical imaging is one of the difficulties. The algorithm need to be capable of spotting tiny irregularities and offering justifications for its forecasts. Researchers are aiming to create explainable AI models that might provide light on how YOLOv3 makes decisions.[23]

Robotic Object Detection:

Numerous industries, including manufacturing, logistics, and healthcare, use robots. Robots can be helped in their work by using YOLOv3 to identify things in real-time. For instance, YOLOv3 may help robots pick up and arrange products on a production line in the proper location by detecting and identifying them. Robots can benefit from YOLOv3's path planning and obstacle detection capabilities to help them navigate.

The requirement for resilience and adaptability while employing YOLOv3 in robots is one of the difficulties. The model should be able to accommodate changes in object form, texture, and illumination. [25]

Challenges in Object Detection using YOLOv3:

The existence of occlusions is one of the main obstacles in object recognition with YOLOv3. When one item in a picture completely or partially obscures another, this is known as an occlusion. It can be challenging for YOLOv3 to precisely detect and localise items in a picture when there are occlusions. Furthermore, YOLOv3 might incorrectly classify an occluded object as a component of another object. Researchers have suggested a number of methods to address this issue, including the use of context-based data and the incorporation of past knowledge about the items in the image.[26]

The existence of tiny items presents another difficulty in object recognition with YOLOv3. YOLOv3 may have trouble recognising little items like writing, small animals, or traffic signs because it is designed to detect medium to big objects. Due to YOLOv3's limited grid cell size and potential for missing tiny items that are in between the grid cells, this restriction exists. In the part after this, we'll go into more depth about this restriction.[5]

YOLOv3 performance issues:

Although YOLOv3 is renowned for its speed and accuracy, it may still experience performance problems when used for real-time applications or on outdated hardware. In comparison to other object identification techniques, YOLOv3 has comparatively high computing needs. This is because YOLOv3 employs a very memory- and processing-intensive deep neural network with several layers. Additionally, YOLOv3 must process each image separately, which can be costly computationally for applications that require real-time processing or large datasets.[10]

Researchers have suggested a number of optimisation strategies, including model compression, pruning, and quantization, to solve these performance difficulties. Model compression entails deleting layers from the neural network that are redundant or superfluous in order to shrink its size. In order to lower the computational load on the neural network, pruning entails eliminating the least significant connections. To save memory, quantization requires decreasing the accuracy of the weights and activations in the network.[18]

YOLOv3's limitations in detecting small objects:

Because of its constant grid cell size, YOLOv3 may have trouble detecting tiny objects, as was already described. In order to anticipate the object class and bounding box for each grid cell, YOLOv3 splits the input picture into a predetermined number of grid cells.[20] This method, however, could overlook tiny things that fit in the gaps between the grid cells or are smaller than a single grid cell. Researchers have suggested a number of methods, such as the use of a multi-scale strategy and a pyramid of feature maps, to get around this constraint.[3]

The multi-scale strategy entails merging the predictions from several scales after processing the input image at various resolutions. By using this method, YOLOv3 is able to find minuscule items that could go unnoticed at a single scale. Utilising a hierarchical group of feature maps with various resolutions is

part of the pyramid of feature maps approach. The predictions from each feature map, which is made to recognise objects of various sizes, are merged to generate.[19]

Future Directions and Recommendations:

Potential YOLOv3 Algorithm Improvements

i. Fusion of Multiple Scale Features:

Multi-scale feature fusion, which can assist increase the accuracy of object recognition, is one potential upgrade for YOLOv3. Multi-scale feature fusion is the process of extracting features from several picture scales and combining them to create a more complete representation of the image. This may be accomplished by concatenating features from various scales using skip connections or several convolutional layers with various kernel sizes.[18]

ii. Redesigned Anchor Box:

YOLOv3 detects objects using predefined boxes called anchor boxes. By creating better anchor boxes that are more suited to the items being recognised, object detection accuracy may be raised. One method is to organise items according to their sizes and forms using a clustering technique, and then utilise the resulting clusters to build anchor boxes. As a result, the anchor boxes will be a better representation of the items being recognised, increasing the object detection algorithm's precision.[18]

iii. Mechanisms of Attention:

By concentrating on the most important elements of an image, attention processes can assist increase the accuracy of object recognition. According to how relevant they are to the objects being recognised, certain portions of a picture are given weights by attention processes. To do this, attention modules may be added to the YOLOv3 architecture. By concentrating on the most important elements of a picture, these modules can increase the accuracy of object recognition.

iv. Contextual Details:

By supplying more details about the items being recognised, contextual information can also assist object detection become more accurate. Information regarding the environment, the scene, or the backdrop might be considered contextual information. Utilising contextual data to fine-tune the bounding boxes of the detected objects is one method. This may be done by extracting contextual data from the image using a contextual module, utilising that data to improve the bounding boxes of the recognised items, and so on.[21]

v. More accurate training data

Improved training data can also increase object detection's accuracy. This may be achieved by utilising more varied datasets, generating additional training data through data augmentation approaches, or utilising data from many areas. The YOLOv3 algorithm can learn to recognise objects more precisely and robustly by utilising richer training data.[1]

Research and development priorities:

i. Object detection in video:

Video object identification is one area that needs more investigation and advancement. The single-image object identification functionality of YOLOv3 may be expanded to include video object recognition by separately processing each frame of a movie.[16] This method does not account for the temporal links between frames, hence it might not be the best one for video object recognition. New algorithms are thus required that can recognise video objects by simulating the temporal interactions between frames.[15]

ii. Tracking Objects in Real-Time:

Another area that needs further investigation and improvement is real-time object tracking. Finding and following objects in a video clip is the process of object tracking. YOLOv3 has real-time object detection capabilities but lacks object tracking.[5]

Conclusion:

The Literature Review's Implications for Object Detection Research

The YOLOv3 algorithm has significantly improved object detection tasks, as shown in the literature review. Its cutting-edge performance across a variety of domains makes it an attractive algorithm for academics and professionals working in the fields of artificial intelligence and computer vision.[28]

A number of elements, including YOLOv3's end-to-end architecture, multi-scale prediction, and feature pyramid network, are responsible for its success. These elements enable YOLOv3 to better recognise tiny items, which is a typical issue in object identification tasks, and to detect objects of various sizes.[28]

Last Words on the YOLOv3 Algorithm:

The YOLOv3 method has become one of the top algorithms for object identification tasks, and several studies have shown that it performs at the cutting edge. It is a potential algorithm for applications including surveillance, autonomous cars, and robots because to its real-time processing speed and excellent accuracy.

YOLOv3, nevertheless, is not without its drawbacks. Particularly when recognising tiny things, its accuracy still needs to be enhanced. The algorithm's high computational and memory requirements might also be a barrier for some applications.[29]

Chapter-3 System Design and Development

3.1 Introduction

Since we're looking for flexible ways to handle and enhance this activity, a coordinated approach will be the best course of action.

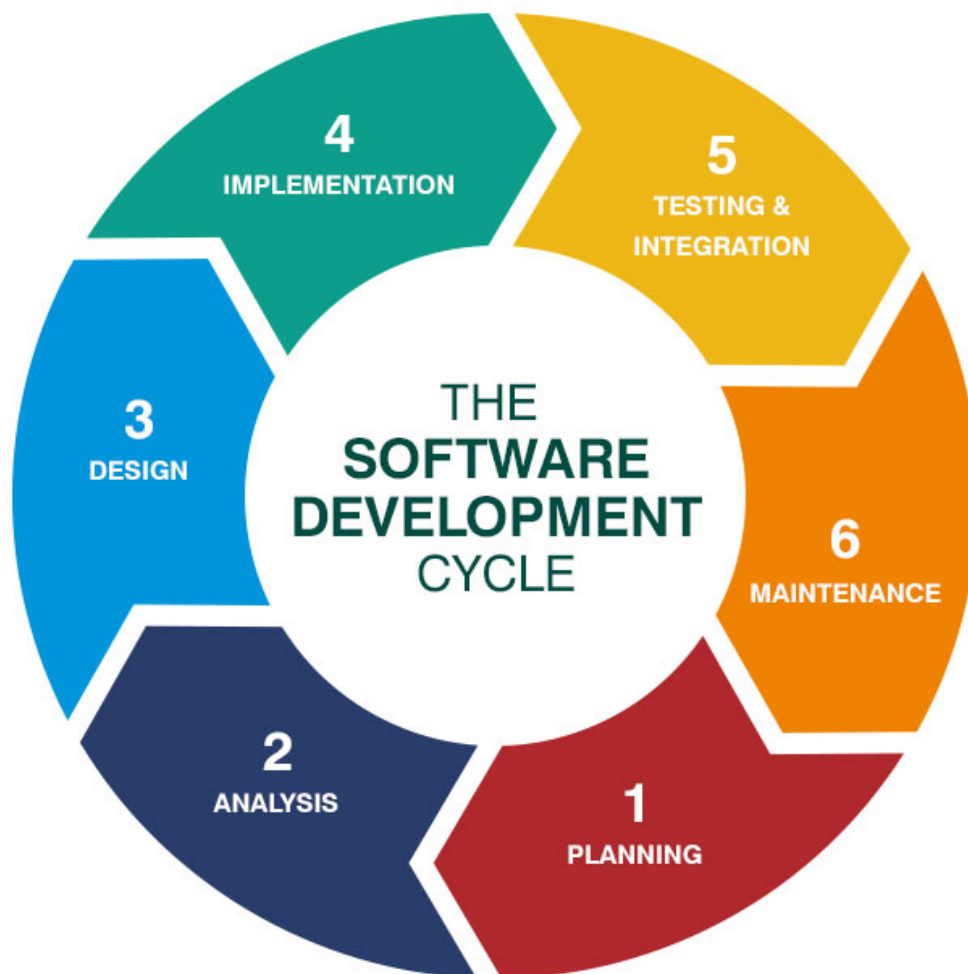


Fig-3.1- Software Development and Life Cycle

(source:<https://bigwater.consulting/2019/04/08/software-development-life-cycle-sdlc/>)

Working programming is conveyed regularly (weeks instead of months). Eye to eye discussion is the best type of correspondence. Close day to day collaboration

between money managers and designers. Nonstop consideration regarding specialised greatness and great plan. Customary variation to evolving conditions.

Indeed, even late changes in prerequisites are invited.

Project Schedule and Gantt Chart

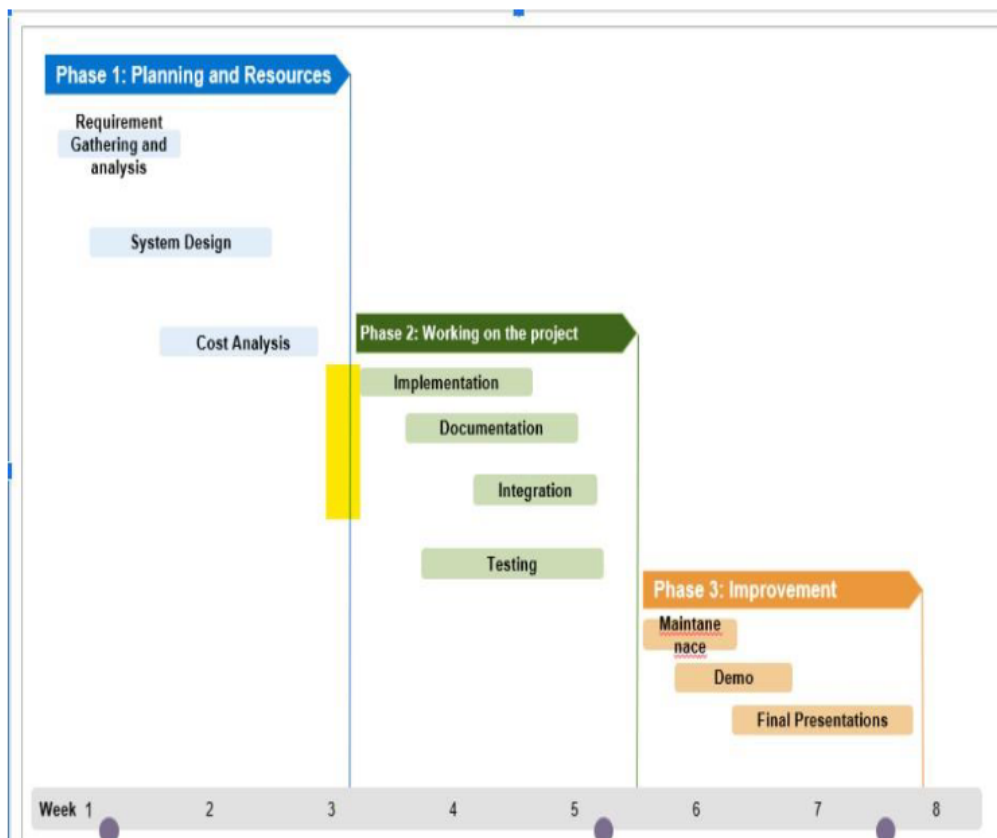


Fig-3.2- Gantt chart

3.2 Conditions

Depending on the particular project and its specifications, the hardware and tools required for web development may also change. However, some popular equipment and tools are as follows:

- Computer: A machine with the processing and memory capabilities required for handling web development tasks.
- An operating system that is compatible, such as Windows, macOS, or Linux.
- Text Editor or Integrated Development Environment (IDE): To write and modify code, use a text editor or IDE like Visual Studio Code, Atom, or Sublime Text.
- Web browser: A web browser to test and preview the website, such as Google Chrome, Mozilla Firefox, or Safari.
- Version Control Software: To monitor code changes and work with other developers, use a version control programme like Git.
- Adobe Photoshop or Sketch are two examples of graphics editing software that may be used to design and alter visuals for a website.
- An FTP/SFTP client, like FileZilla, is used to transfer files between a local computer and a web server.
- Testing and Debugging Tools: To find and repair coding mistakes, use testing and debugging tools like Chrome Developer Tools, Firebug, or Fiddler.
- Server-Side Technologies: These include server-side programming languages like PHP, Ruby, Python, or Node. Depending on the needs of the project, js may also be needed.

The process of object detection is intricate and requires both hardware and software technologies. Some of the instruments and programmes frequently used in object detection are listed below:

- Cameras or video cameras that take pictures or movies of the items to be recognised are known as image/video capture devices.
- Annotation Tools: To annotate the pictures with bounding boxes, masks, or polygons, use programmes like LabelImg, RectLabel, or VGG Image Annotator (VIA).
- Object Detection Frameworks: Frameworks for building and training object detection models include PyTorch Detection, TensorFlow Object Detection API, and YOLO (You Only Look Once).
- Pre-trained Models: Object identification models can start with pre-trained models like COCO, ImageNet, or Open Images.
- Tools for enhancing data include Albumentations and imgaug, which apply transformations including flipping, rotating, and scaling on the training data.
- Tools for training and evaluating object detection models include TensorFlow Model Garden and PyTorch Lightning.
- Cloud computing services are used to train models on big datasets and draw conclusions from real-world data. Examples of these services are Amazon Web Services (AWS), Google Cloud Platform (GCP), and Microsoft Azure.
- Libraries that provide image processing and computer vision techniques for object detection include OpenCV, Dlib, and scikit-image.

The Design and Development for the website.

The Tech stack used here

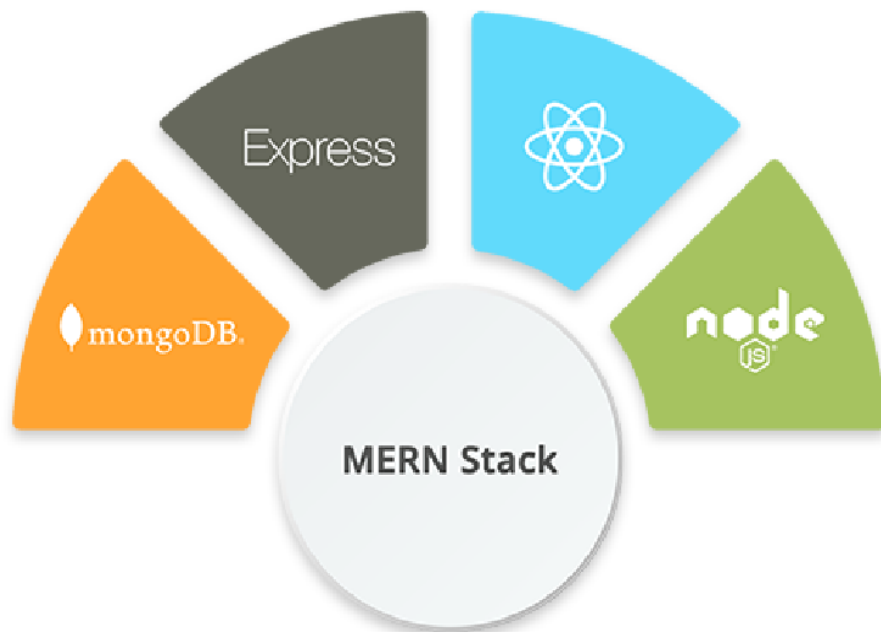


Fig: 3.3: Tech Stack used

(source: <https://www.webhoppers.com/top-mern-stack-development-companies-in-india>)

The Explanation in brief:

1) Mongo DB:

A well-liked NoSQL database, MongoDB is frequently included into the MERN stack for creating full-stack online applications. With the help of JavaScript and other computer languages, it is simple to access and change the data that MongoDB stores in documents that resemble JSON.

MongoDB is frequently used as the database layer in the MERN stack to store and manage the application data. For web applications, MongoDB offers a number of advantages, including:

- **Scalability:** MongoDB is made to expand horizontally across several servers, making it simple to manage massive amounts of data and heavy traffic.

- Flexibility: MongoDB can store data in a flexible schema-less manner that is perfect for dynamic web applications since it is a document-oriented database.
- Performance: MongoDB can handle sophisticated queries and real-time data changes and is optimised for excellent performance.
- Seamless integration of MongoDB with Node.js and other MERN stack technologies facilitates the development and deployment of online applications.
- Developers often utilise the MongoDB Node.js driver or Mongoose, a well-liked object data modelling (ODM) module for MongoDB, when creating a MERN stack application using MongoDB. The CRUD (Create, Read, Update, Delete) actions on the application data may be carried out by developers using these tools, which offer a simple interface for dealing with the database.
- All things considered, MongoDB is a strong and adaptable database solution that is perfect for usage with the MERN stack. It gives programmers the resources they need to create web applications that are quick, scalable, and adaptable.

2) Express:

Express is a well-known open-source Node.js web application framework that offers a comprehensive range of functionality for developing online apps and APIs. Express is a well-liked option for programmers creating Node.js web apps since it is designed to be straightforward, adaptable, and extendable.

Express offers a variety of essential characteristics, such as:

- Middleware: Express enables programmers to create functions that may be used to process incoming HTTP requests and carry out operations like logging, authentication, and error handling.
- Express's robust routing system enables developers to design routes for dealing with various HTTP methods and URLs.
- Template engines: Express supports a number of template engines that let developers create dynamic HTML content depending on information from the server, including EJS, Pug, and Handlebars.

- In order to handle failures and exceptions in a consistent and predictable manner, Express has built-in error handling middleware.
- Security: To assist developers in creating safe online applications, Express has built-in security capabilities including content security rules and cross-site scripting (XSS) protection.
- Because Express is compact and adaptable, programmers may use it to create a variety of online applications, from tiny single-page apps to expansive APIs and web services. Furthermore, because Express is based on Node.js, it takes advantage of its asynchronous, event-driven features to deliver quick and scalable online applications.

3) React Js:

A well-liked open-source JavaScript toolkit for creating user interfaces, is third. React is a popular framework for creating contemporary online apps since it was created by Facebook and is intended to be declarative, effective, and simple to use.

React functions by disassembling the user interface into manageable, reusable components that are simple to control and alter with JavaScript. Only the essential components are updated when data changes thanks to React's effective usage of a virtual DOM (Document Object Model) to manage updates and create the user experience.

React's essential characteristics include things like:

- Reusable components: React makes it simple to construct scalable and maintainable apps by enabling developers to design reusable components that can be used across numerous pages or applications.
- React employs a virtual DOM to rapidly handle updates and render the user interface, making sure that only the essential components are updated as data changes.
- JSX: React makes use of JavaScript XML (JSX), a syntactic extension that enables programmers to write HTML-like code in JavaScript files, simplifying the creation and management of the user interface.
- Data goes from the parent component to its child components in a single direction when using a unidirectional data flow, which is what

React employs. Data management is made simpler as a result, and the user interface is kept current with the status of the programme.

- speed: React is well optimised for speed, making it a wonderful option for creating online apps that are quick and responsive.

Overall, React is a strong and adaptable toolkit that gives programmers the resources they need to create online applications with scalable and maintainable user interfaces. For creating cutting-edge online apps, it's a fantastic option thanks to its popularity and vibrant community support.

4) Node.js:

Node.js is an open-source, cross-platform runtime environment for JavaScript that enables developers to create server-side JavaScript applications. Node.js offers an event-driven, non-blocking I/O architecture that is optimised for developing scalable and high-performance applications. It is built on top of the Chrome V8 JavaScript engine.

Node.js has a number of important features, including:

- Asynchronous programming is possible with Node.js because to its event-driven, non-blocking I/O paradigm, which enables the processing of several requests concurrently without delaying the processing of other requests.
- Cross-platform compatibility: Node.js is a flexible platform for developing server-side applications since it works on a variety of operating systems, including Windows, macOS, and Linux.
- NPM: The Node Package Manager (NPM), a potent package manager included with Node.js, enables developers to quickly install, update, and manage third-party packages and libraries.
- Scalability: Node.js is made to be extremely scalable, having the capacity to manage many connections and requests at once.
- Simple to learn: Because Node.js employs JavaScript as its primary programming language, web developers may easily switch to utilising it to create server-side applications.

Many businesses and organisations utilise Node.js to create a variety of server-side applications, from little web apps to massive enterprise systems. Web servers, real-time applications, microservices, and APIs are a few of the common Node.js use cases.

Website functionality:

A straightforward JuitOLX website generally offers the following fundamental features:

- User registration and login: Visitors to the website can register an account by providing an email address or a social network login. Users may log in to the website after registering to access their accounts and manage their listings.
- Items may be listed on the website by users, who can include information about them such as name, description, pictures, price, and location.
- Items may be found on the internet by using a variety of search parameters, including category, region, and price range.
- Item filtering: Users may narrow down search results based on parameters including location, price range, and item condition.
- Users who have demonstrated interest in certain goods may get updates or new items that meet their search criteria via alerts from the website.
- User reviews: Based on their experiences with purchasing or selling things on the website, users may be able to provide reviews and ratings for other users on the platform.
- Payment integration: The website could include connection with different payment channels, enabling customers to safely pay for the goods they wish to purchase.

A straightforward JuitOLX website's primary purpose is to offer a platform for people to conveniently and securely purchase and sell products online. The aforementioned features assist in this process and make it simpler for users to seek for things, connect with other users, and conduct transactions safely and effectively.

Website user interface:

A straightforward JuitOLX website's user interface (UI) generally consists of the following components:

- Navigation menu: The navigation menu is often seen at the top of a website and offers access to various parts, including user profiles, item categories, and search capabilities.
- The search bar is often prominently displayed on the homepage and enables visitors to look for products using a variety of search parameters, including keywords, category, and location.
- The name, description, image, price, and location of each item are shown in a grid or list format on the website's main part. Users can click on certain listings to access the item's full description
- Filters: The website could include filters that let visitors narrow down the results of their searches based on particular standards like price range, location, and item condition.
- User profiles are available to registered users, who may use them to see and edit their listings, messages, and other account preferences.
- Logging in or registering with the website is required to use certain functions, such as publishing new listings or messaging other users. Users who are not signed in are prompted to do so.

A straightforward JuitOLX website's user interface is designed to be simple to use and intuitive, with prominent search and filtering options, concise item listings, and an organised user profile and message system. Making it as simple as possible for users to locate the things they are searching for, interact with other users, and carry out transactions safely and quickly is the aim.

Database Design:

The database design for a basic OLX website would be as follows, assuming that the website just includes these tables:

- Buyer: The name, email address, password, and other pertinent information regarding customers who have registered on the website are all kept in this table.
- Seller: The names, email addresses, passwords, and other pertinent information of the sellers who have registered on the website are kept in this table.
- Login: This table contains information on the username and password used to log in for both buyers and sellers.
- Users' names, email addresses, passwords, and other pertinent information are all kept in this table for each registered user.
- The name, description, image, price, location, and other pertinent information about the things that sellers have put for sale on the website are all stored in this table under the heading "Add Product."
- Remove Product: The name, description, image, price, location, and other pertinent information about the products that sellers have taken from the website are all stored in this table.

The login table contains data on users' login credentials, whereas the buyer and seller tables provide particular information about the people who have registered with the website. A more comprehensive list of all registered users is available in the user table. While the remove product table contains data about things that have been taken from the website, the add product table contains data about the items that sellers have offered for sale there.

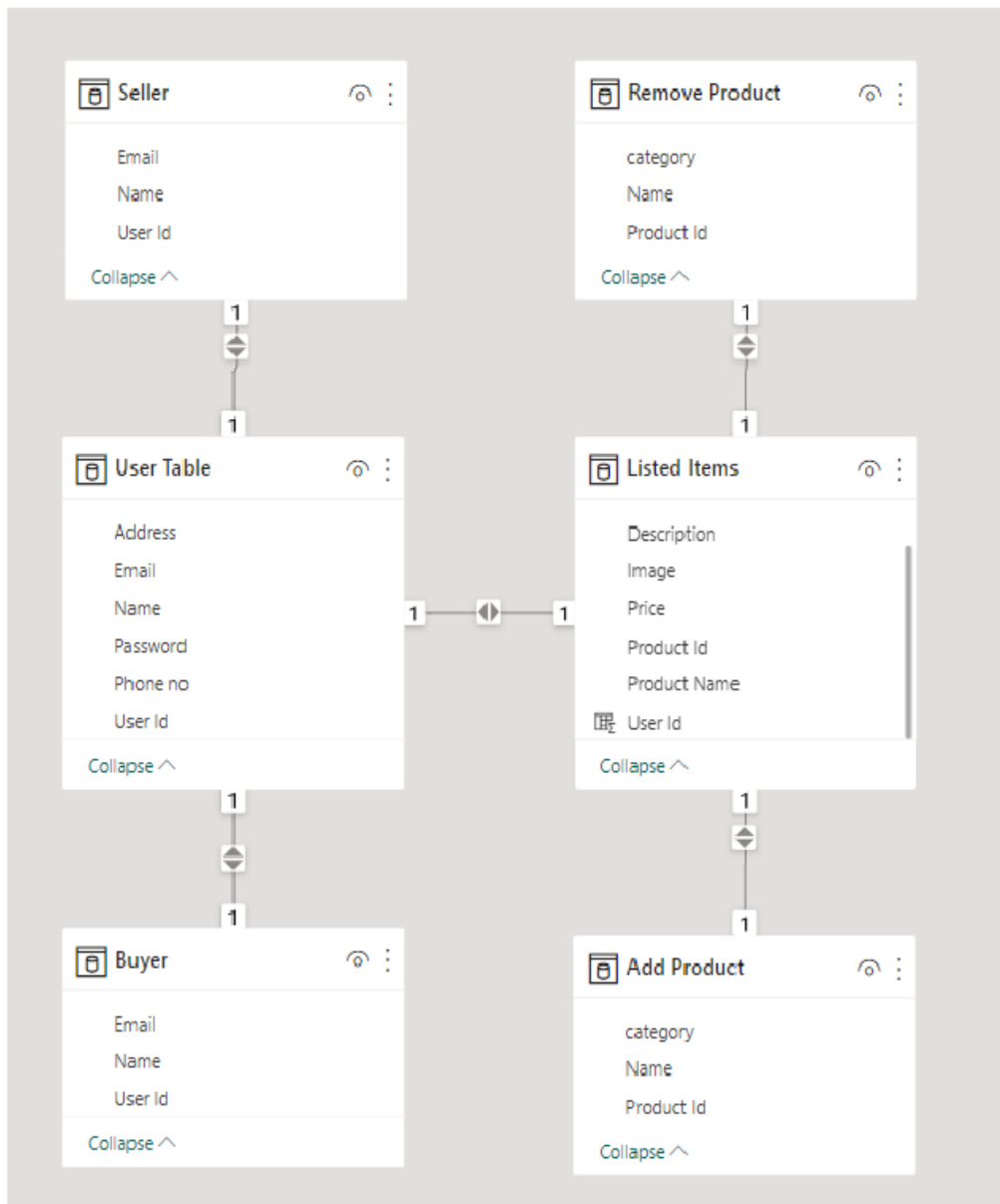


Fig 3.4: Database design

Here, the "User id" column on the User side and the "Product Id" column on the Product side are used to create the linkages, and the "User id" column connects the two sides.

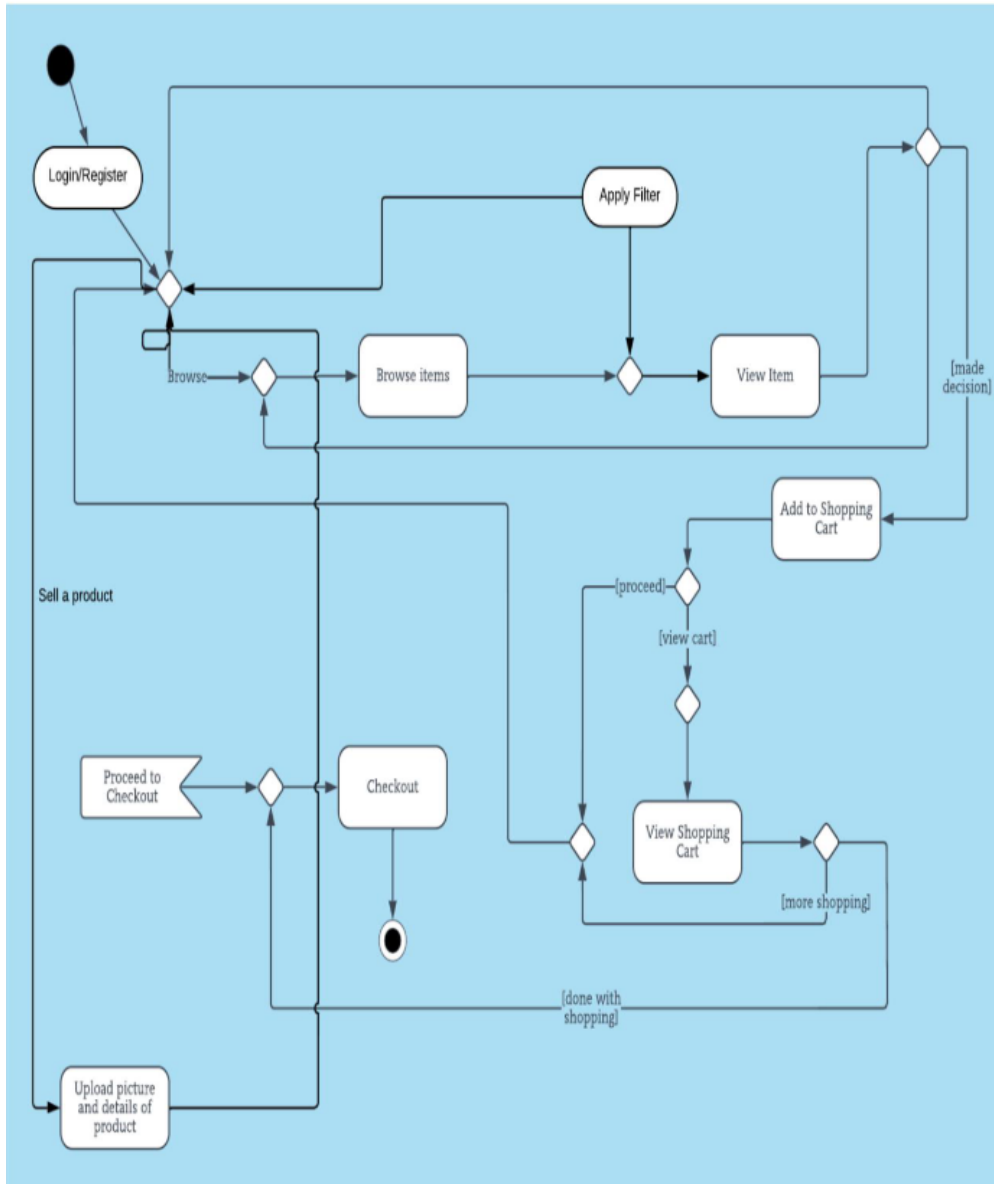


Fig 3.5: Website flowchart or action diagram

The website's flow:

A example flow of a straightforward OLX website with the provided pages might be as follows:

- Home Page: The website's home page would serve as its landing page. It would link to the login and registration pages and have a search bar. Users might either explore the offered categories or do keyword searches for items.
- Users who are not currently logged in will be sent to the login or signup page. To register, new users would need to supply their name, email address, password, and other pertinent information. Users logging in for the second time would input their username and password.
- User profile: After logging in, users are taken to their profile page, where their name, profile photo, and other pertinent information are shown.
- Users have the option to change the information on their profiles, add goods to sell, and examine both their recent and previous transactions.
- Buy and Sell Page: Customers may browse or do searches on the buy and sell page to find goods that are offered for sale on the website. While customers may get in touch with merchants directly to haggle over a price or request additional information, sellers could add new things to offer.
- Checkout Page: After deciding to buy a product, a customer is sent to the checkout page, where they enter their billing and shipping information and complete the transaction.

The website would generally flow from the main page to the login or registration page, to the user profile page, to the buy and sell page, and then to the checkout page for finalising transactions.

Models for object detection/phase:

A computer vision technique called object detection is used to find and categorise things in an image or video. There are several distinct object detection models, each with unique advantages and disadvantages.

Some of the most well-liked object detection models are listed below:

- YOLO (You Only Look Once): Known for its accuracy and quickness, YOLO is a real-time object detection model. To estimate bounding boxes and class probabilities for each cell, YOLO splits each picture into a grid.
- Faster R-CNN (Region-based Convolutional Neural Network) is a well-known object identification model that generates prospective item placements using a region proposal network, then utilises a second network to improve these recommendations and categorise the objects.
- A single neural network is used by SSD (Single Shot Detector), a real-time object identification model, to forecast bounding boxes and class probabilities for all objects in an image.
- RetinaNet is a cutting-edge object identification model that solves the issue of class imbalance in object detection by utilising a revolutionary focus loss function.
- Masked R-CNN In order to forecast pixel-level masks for each item in addition to bounding boxes and class probabilities, Mask R-CNN, an extension of Faster R-CNN, adds a third branch.

These are only a handful of the numerous object detection models that are available. The precise needs of the application, including elements like speed, accuracy, and available computer resources, will determine the model to be used.

Real-time object identification approach known as YOLO (You Only Look Once) is well-liked for its quickness and precision. In order to anticipate bounding boxes and class probabilities for each cell, YOLO divides each picture into a grid.

An overview of the YOLO architecture is provided below:

- Layer of input: This layer feeds the network with the input picture.
- The task of extracting features from the input picture falls to the CNN layers.
- The detection layers are in charge of foretelling each cell's bounding boxes, confidence scores, and class probabilities.
- The final set of bounding boxes, confidence ratings, and class probabilities are generated by the output layer.

Here is a general explanation of how YOLO functions:

- picture input: YOLO splits the supplied picture into a grid of cells.
- Feature extraction: YOLO extracts features from each cell of the input picture using a convolutional neural network (CNN).
- YOLO produces a predetermined number of bounding boxes and associated confidence ratings for each cell, as well as class probabilities for every item contained inside the bounding boxes. The class probabilities describe the likelihood that the item belongs to a certain class (e.g., person, automobile, etc.), whereas the confidence score represents the degree of certainty that the cell contains an object.
- Non-maximum suppression: To get rid of overlapping boxes with lower confidence ratings, YOLO uses the non-maximum suppression (NMS) method on the predicted bounding boxes.
- A set of bounding boxes with corresponding confidence scores and class probabilities make up the final product of YOLO.

One of YOLO's main benefits is its speed and real-time performance, which makes it a popular option for applications like robotics, self-driving cars, and video surveillance.

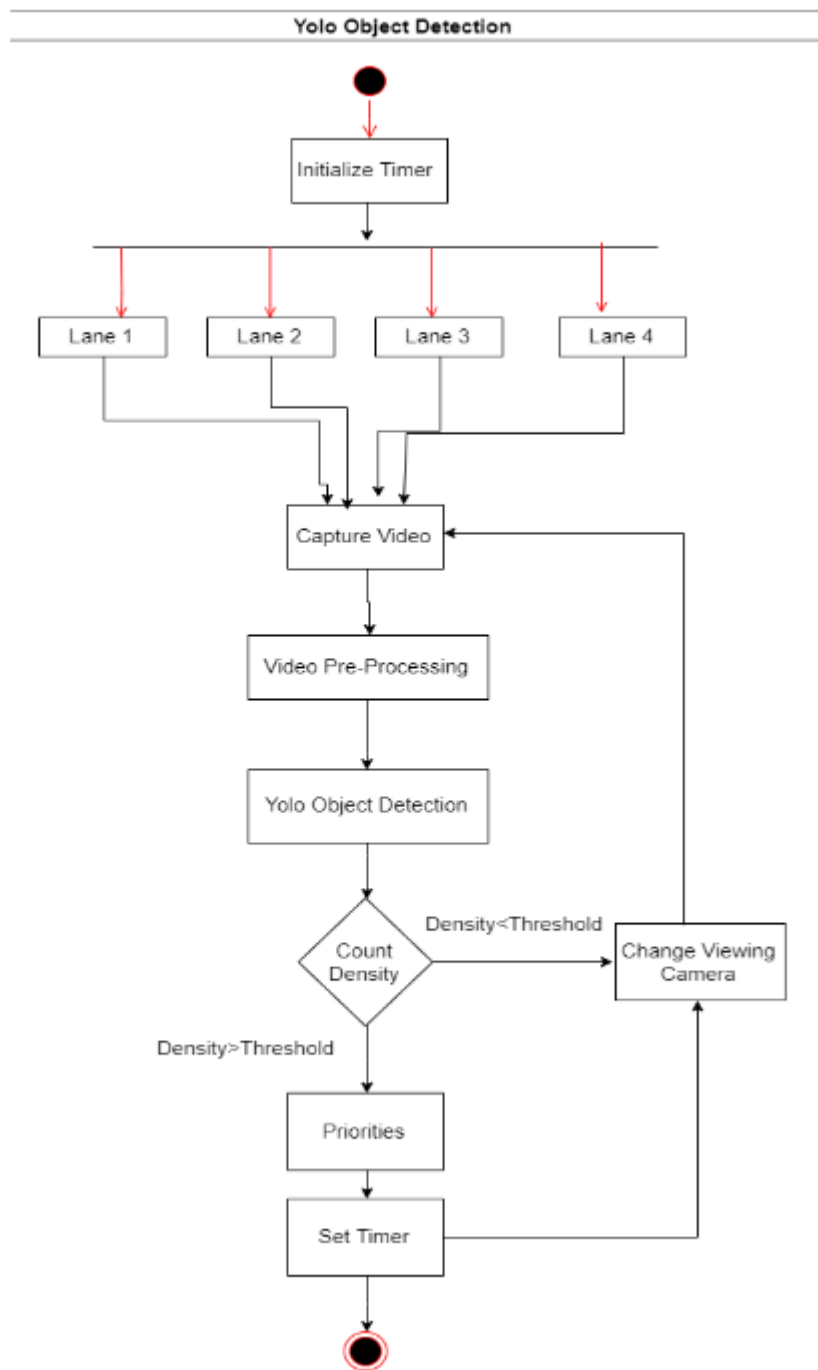


Fig. 3.6 The Architecture of YOLO

Chapter 4

Experiment and Result Analysis

Following the website development in these steps:

1. Install Node.js:

by downloading it from the official website and installing it. Npm (Node Package Manager), included with Node.js, enables us to install the dependencies and packages our project needs.

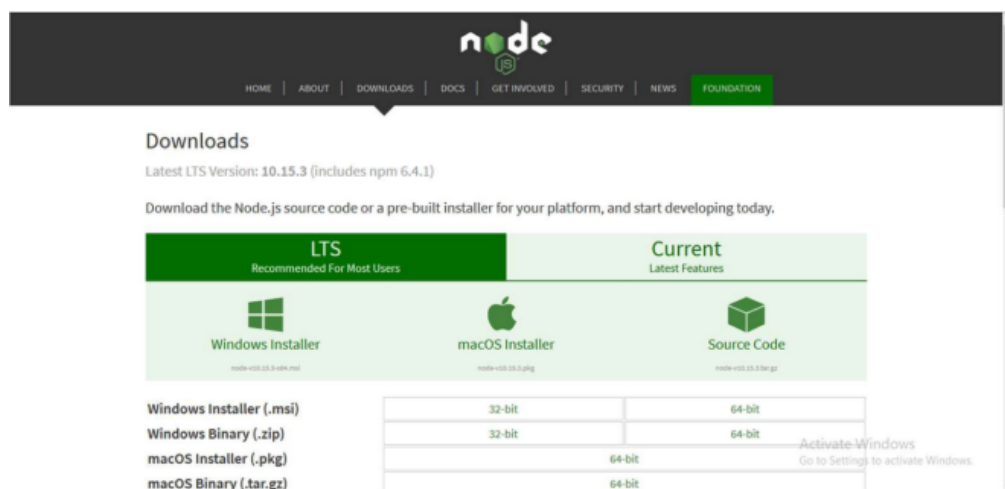


Fig 4.1: Node Download page

2. Start a new project:

For your website, make a new project directory, then use the command line to go there.

3. Start the project:

The command "npm init" will start the project and create a package.json file in the project directory. The metadata for your project and a list of installed dependencies are both contained in this file.

4. Install Express:

In your project directory, type "npm install express" to install Express. The popular Node.js framework Express is used to create web apps.

5. Install React: In your project directory, type "npm install react" to install React. A JavaScript package called React is used to create user interfaces.

6. Configure the server:

Create a new file called "server.js" and configure the Express server in your project. Import the Express module and configure the server to listen for incoming requests in this file.

7. Set up the client:

Create a new directory in your project called "client" to set up the React client. Use the command "npx create-react-app." (with the dot at the end) to build a new React app in this directory. This will create a new React app in the current directory.

8. Establish a connection between the client and the server:

Establish a connection between the client and the server by adding a proxy to the "package.json" file in the client directory. The client and server will be able to communicate as a result.

9. Create and execute the application:

Create the application by executing the "npm run build" command in the client directory. The React app will then have a build that is ready for production. After that, launch the server by typing "node server.js" into the project directory. This will launch the server and open a certain port, allowing users to visit the website.

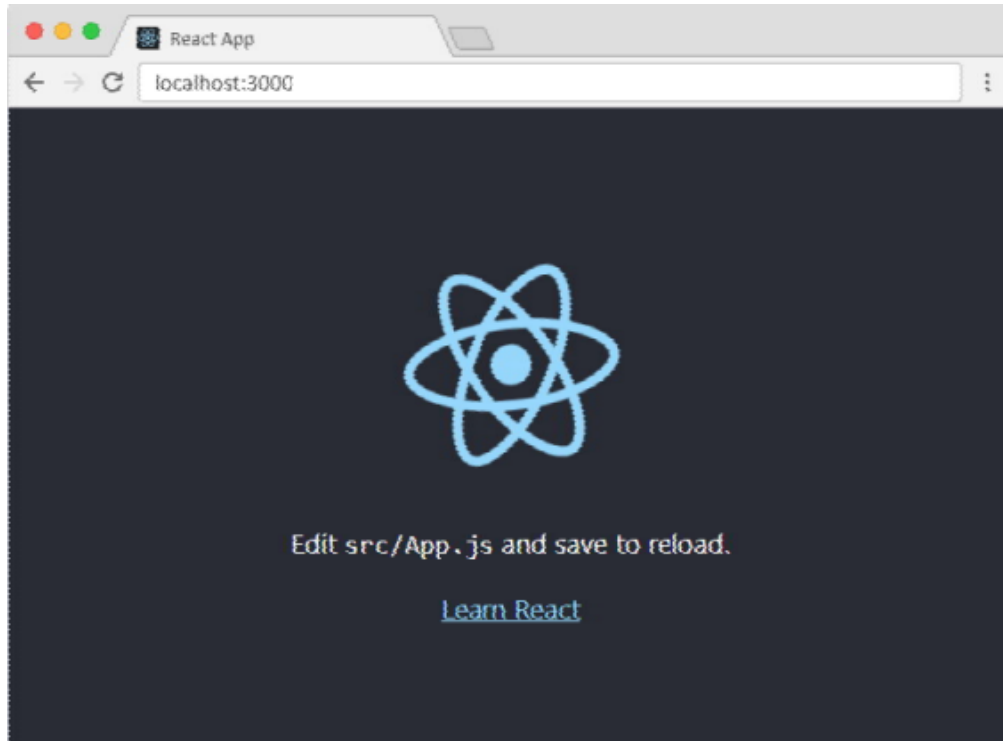


Fig 4.2: Starting React

10. Create and launch the website:

Lastly, create and launch the website by customising the server.js and React app code to meet your needs. Deploy the website to a hosting service once it is ready. public access service or cloud platform

Findings from the website:

OLX

mayankekaghara.me@gmail.com
Email address

.....
Password

Remember me [Forgot password?](#)

[Login](#)

[Don't have an account? Register](#)

Made By Mayank & Aditya

Fig 4.3: Login page

OLX Home category Profile Sell Product

Product Name

Choose Category ▾

Price

Image:

Description

Made By Mayank & Aditya
localhost:3000/olw/ads/5374a7c75a78e1f49b72c5

Fig 4.4: Sell page

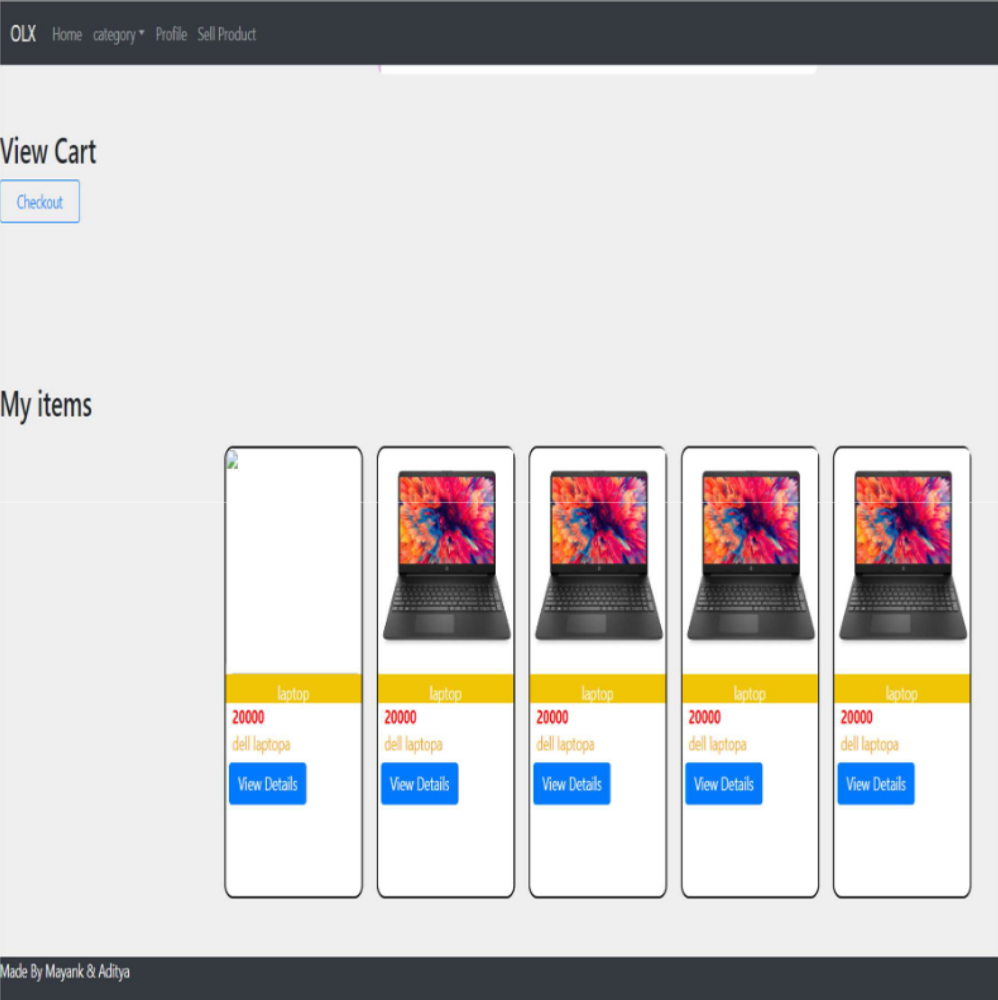


Fig 4.5: Cart page

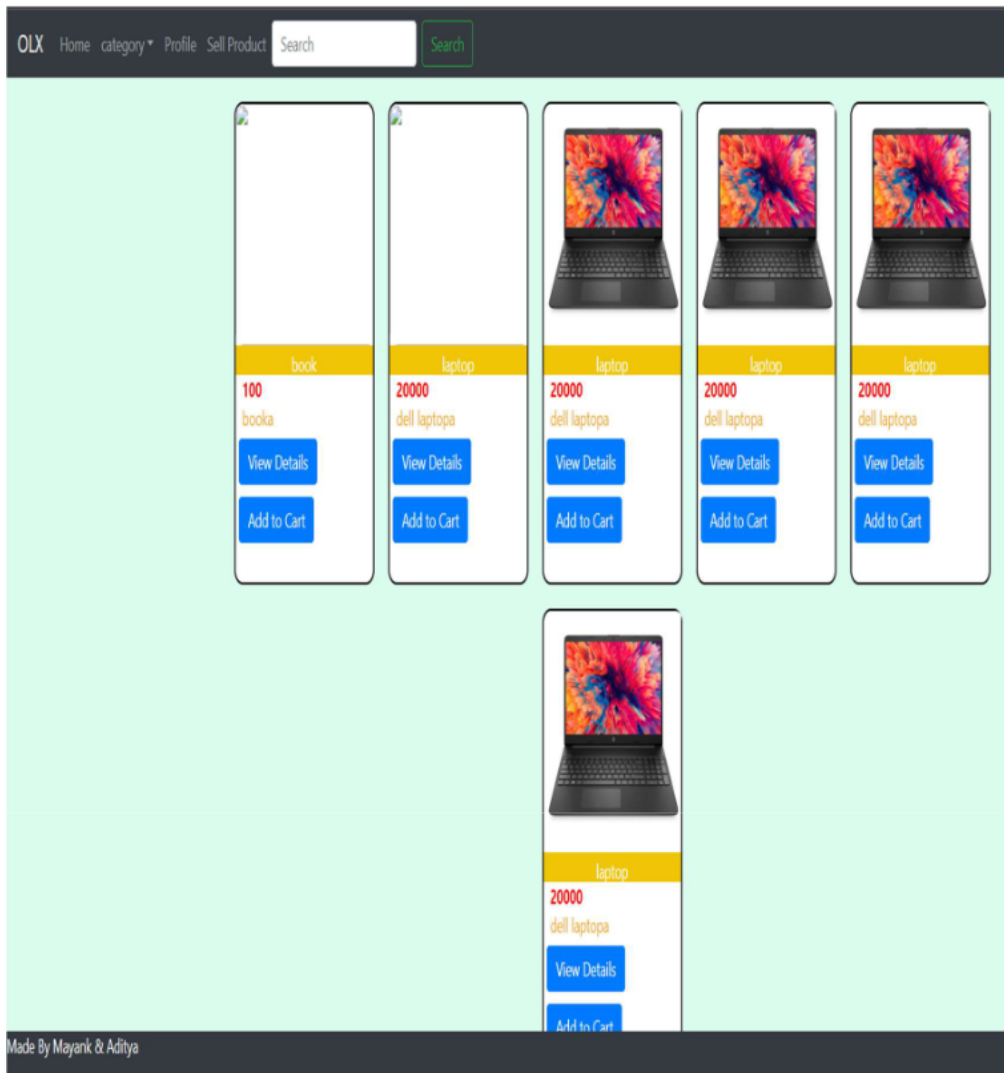


Fig 4.6: Home page

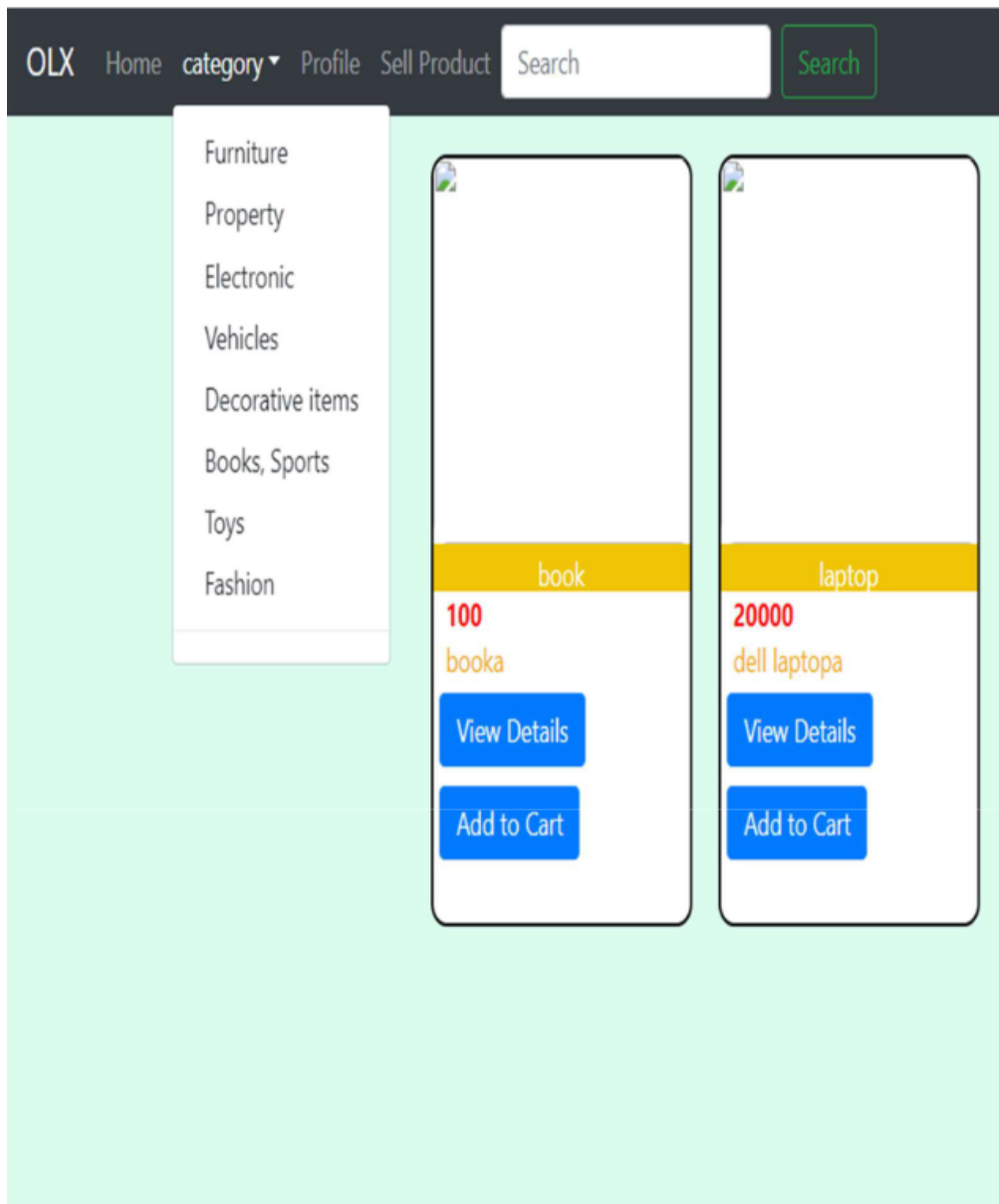


Fig 4.7: Category page

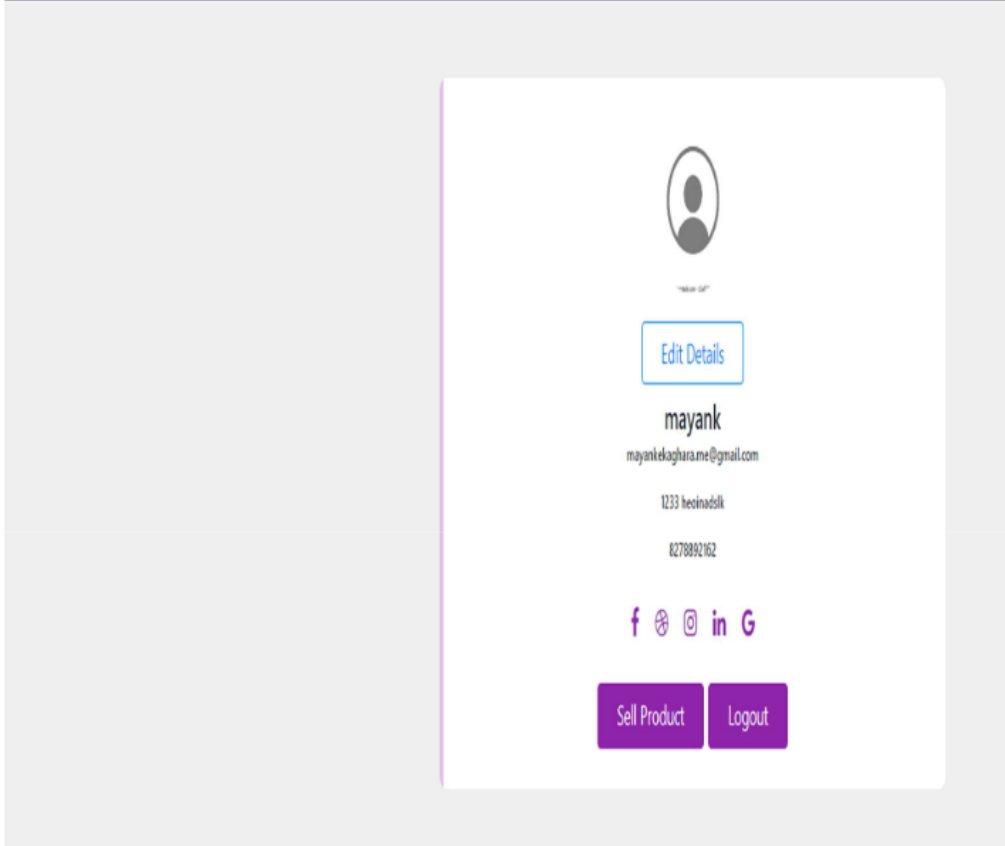


Fig 4.8: Profile Page

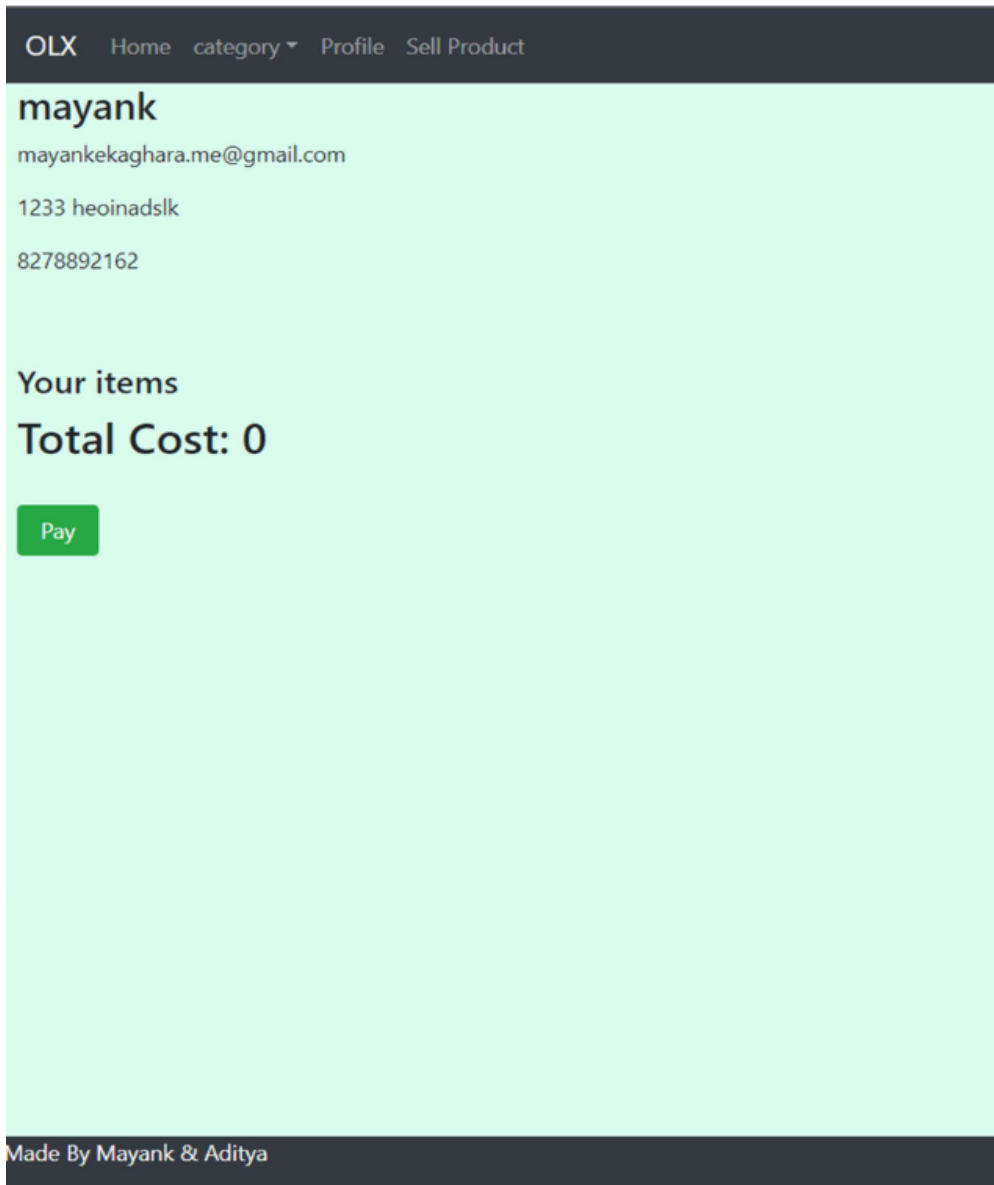
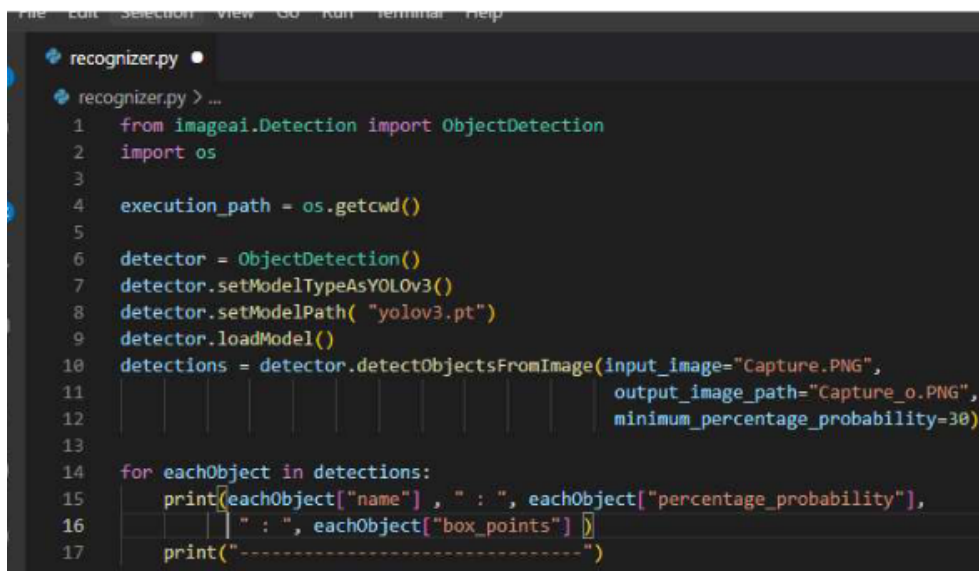


Fig 4.9: Checkout page

How to put the Object detection model into practise:

1. Decide which YOLO version best suits your requirements. There are many YOLO variants, including YOLOv1, YOLOv2, YOLOv3, and YOLOv4, each with a unique architecture and set of performance properties.
2. Save the pre-trained weights for the chosen YOLO version to your computer. On many websites, including the official YOLO website, GitHub repositories, or other sources, you may find pre-trained weights for YOLO models.
3. Install the prerequisites required to run the YOLO model. Installing deep learning frameworks like TensorFlow or PyTorch as well as additional dependencies like OpenCV, NumPy, and Matplotlib is required.
4. Fill the YOLO model with the pre-trained weights. The model's parameters may be initialised using the pre-trained weights, and it can then be fine-tuned using data from your dataset.
5. Apply the YOLO model to your movies or pictures. You may use the model to identify items in fresh pictures or videos after it has been loaded and tweaked. To create bounding boxes around the identified objects, you must first give the model with the input data, preprocess it, and then utilise the model's output.

A screenshot of a code editor window showing a Python script named 'recognizer.py'. The code imports 'ObjectDetection' from 'imageai.Detection' and 'os'. It sets the execution path to the current directory, initializes an 'ObjectDetection' detector, sets the model type to 'YOLOv3', and sets the model path to 'yolov3.pt'. The detector is then loaded. The script calls 'detectObjectsFromImage' with 'Capture.PNG' as the input image, 'Capture_o.PNG' as the output image path, and a 'minimum_percentage_probability' of 30. A loop iterates over the 'detections' list, printing the name, percentage probability, and box points for each object.

```
recognizer.py
recognizer.py > ...
1  from imageai.Detection import ObjectDetection
2  import os
3
4  execution_path = os.getcwd()
5
6  detector = ObjectDetection()
7  detector.setModelTypeAsYOLOv3()
8  detector.setModelPath( "yolov3.pt")
9  detector.loadModel()
10 detections = detector.detectObjectsFromImage(input_image="Capture.PNG",
11                                             output_image_path="Capture_o.PNG",
12                                             minimum_percentage_probability=30)
13
14 for eachObject in detections:
15     print(eachObject["name"] , " : ", eachObject["percentage_probability"],
16           " : ", eachObject["box_points"] )
17     print("-----")
```

Fig 4.10: Item recognition code

In the illustration above, we see:

- 'from imageai.Detection import ObjectDetection' imports the ObjectDetection class from the module imageai.Detection. This class offers a simple API for deep learning models that have already been trained to perform object detection.
- 'import os' imports the os module, which gives users a mechanism to communicate with the operating system by accessing files and directories, for example.
- 'execution_path = os.getcwd()': This statement retrieves the current working directory and puts it in the 'execution_path' variable. In a subsequent step, the input and output picture paths will be specified using this variable.
- 'detector = ObjectDetection()' creates an instance of the ObjectDetection class.
- The code 'detector.setModelTypeAsYOLOv3()' changes the model type to YOLOv3. A well-liked object identification model with a reputation for swiftness and precision is YOLOv3.
- The code 'detector.setModelPath("yolov3.pt")' specifies the location of the pre-trained YOLOv3 weights file. The learnt parameters for the model are contained in the weights file.
- The line 'detector.loadModel()' loads the pre-trained YOLOv3 model into memory. The model is currently prepared for use in object detection.
- detections=
detector.detectObjectsFromImage(input_image="Capture.PNG",
output_image_path="Capture_o.PNG",minimum_percentage_probabili
ty="30"); This line conducts object detection on the "Capture.PNG"
input picture and records the objects and their characteristics in the
variable "detections." The path to save the output picture with
bounding boxes generated around the identified items is specified by
the 'output_image_path' option. The minimal degree of confidence
necessary for an item to be regarded as a legitimate detection is
specified by the 'minimum_percentage_probability' option.

- If "for eachObject in detections:" is used: The for loop that iterates over each object in the 'detections' variable that was detected starts with this line.
- "print(eachObject["name"], ":", "percentage_probability", ":", and "box_points")" The name of the identified item, the likelihood that it was detected, and the coordinates of the bounding box surrounding the object are all printed on this line.
- `print("-----")`: To visually separate the output of several items, this line displays a separator.

YOLO Detections Results:

Test Input Image 1:



Fig 4.11: Input image 1

Output Image 1:



Fig 4.12: Output image 1

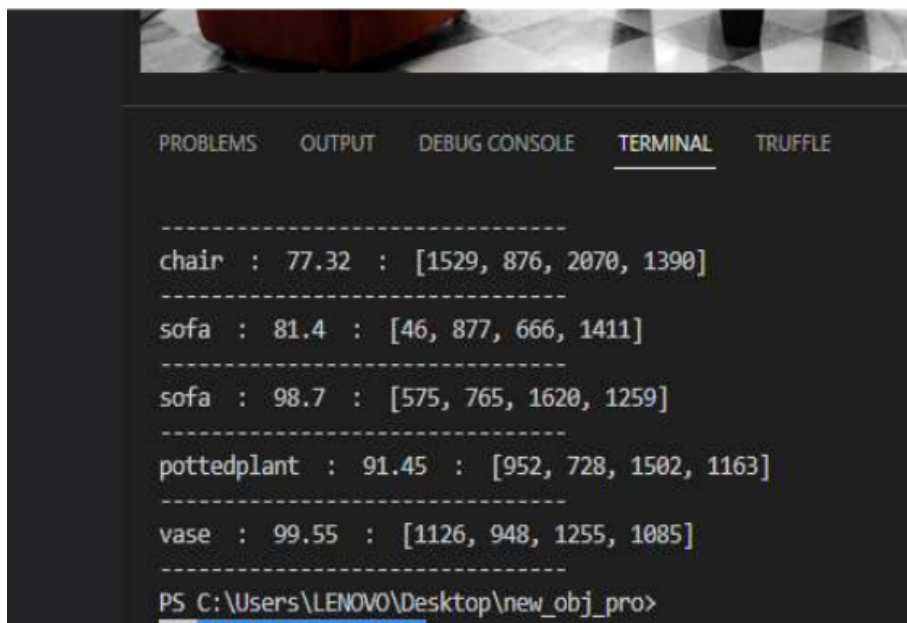


Fig 4.13: Item Recognised 1

Test input image 2



Fig 4.14: Input image 2

Output 2:

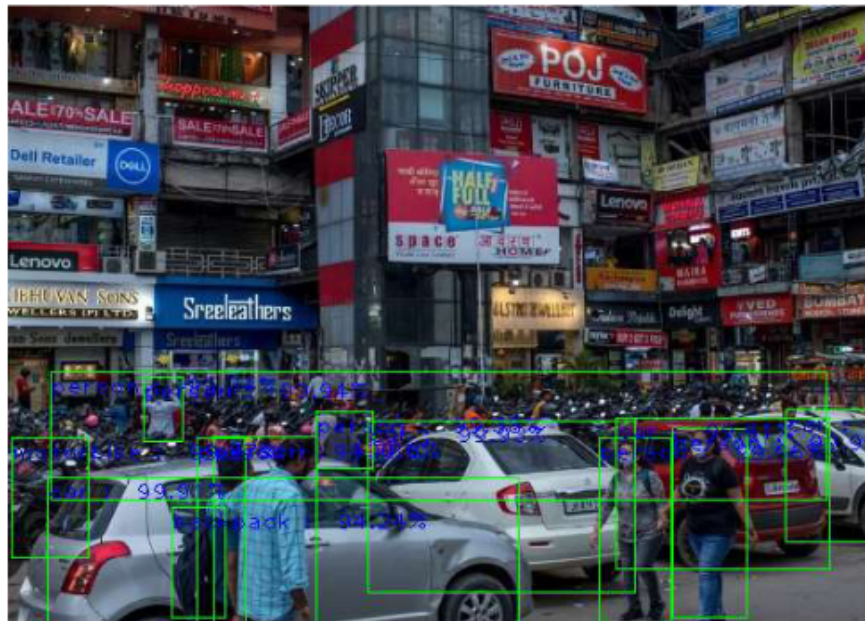


Fig 4.15: Output image 2

```
13 : ".eachObject["box_points"] )
PROBLEMS OUTPUT DEBUG CONSOLE ... Code
NameError: name 'apture' is not defined
[Done] exited with code=1 in 0.214 seconds
[Running] python -u "c:\Users\LENOVO\Desktop\new_obj_pro\recognizer.py"
person : 99.96 : [376, 275, 422, 392]
-----
person : 100.0 : [131, 274, 196, 394]
-----
person : 99.95 : [423, 269, 470, 389]
-----
person : 99.78 : [120, 275, 151, 395]
-----
person : 99.38 : [196, 258, 232, 294]
-----
person : 99.34 : [27, 233, 521, 314]
-----
person : 99.94 : [86, 236, 111, 277]
-----
car : 99.93 : [229, 264, 399, 373]
-----
car : 99.91 : [25, 300, 325, 394]
-----
car : 99.61 : [387, 262, 523, 358]
-----
car : 99.53 : [495, 257, 550, 341]
-----
motorbike : 99.27 : [2, 275, 51, 351]
-----
backpack : 94.24 : [104, 319, 138, 389]
-----
```

Fig 4.16: Recognised image 2

Performance of the Yolov3 object detection model:

A pre-trained YOLOv3 model's accuracy can vary based on a number of variables, including the amount and quality of the training dataset, the model's particular setup, and the type of objects being recognised.

The Darknet framework's pre-trained YOLOv3 model was developed using data from the COCO dataset, which includes pictures of objects in 80 distinct categories. The YOLOv3 model performed quite well for a real-time object identification model on this dataset, with mean Average Precision (mAP) of 57.9 and Intersection over Union (IoU) of 33.1.

YOLOv3 is incredibly quick and precise. YOLOv3 is comparable to Focal Loss in mAP measured at.5 IOU, but it is nearly four times quicker. Additionally, you may easily compromise between accuracy and speed by just altering the model's size; no retraining is necessary!

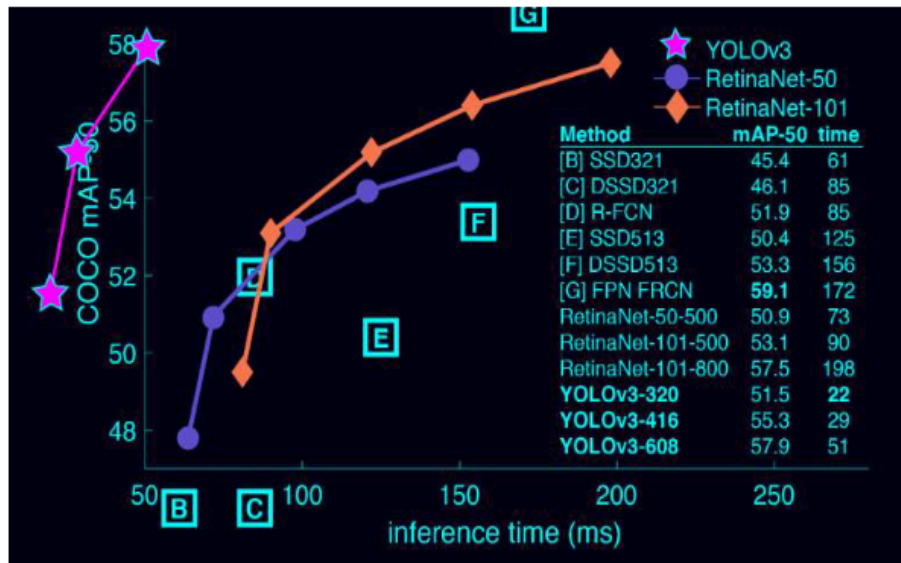


Fig 4.17: Evaluation of several detectors

Model	Train	Test	mAP	FLOPS	FPS	Cfg	Weights
SSD300	COCO trainval	test-dev	41.2	-	46		link
SSD500	COCO trainval	test-dev	46.5	-	19		link
YOLOv2 608x608	COCO trainval	test-dev	48.1	62.94 Bn	40	cfg	weights
Tiny YOLO	COCO trainval	test-dev	23.7	5.41 Bn	244	cfg	weights
SSD321	COCO trainval	test-dev	45.4	-	16		link
DSSD321	COCO trainval	test-dev	46.1	-	12		link
R-FCN	COCO trainval	test-dev	51.9	-	12		link
SSD513	COCO trainval	test-dev	50.4	-	8		link
DSSD513	COCO trainval	test-dev	53.3	-	6		link
FPN FRCN	COCO trainval	test-dev	59.1	-	6		link
Retinanet-50-500	COCO trainval	test-dev	50.9	-	14		link
Retinanet-101-500	COCO trainval	test-dev	53.1	-	11		link
Retinanet-101-800	COCO trainval	test-dev	57.5	-	5		link
YOLOv3-320	COCO trainval	test-dev	51.5	38.97 Bn	45	cfg	weights
YOLOv3-416	COCO trainval	test-dev	55.3	65.86 Bn	35	cfg	weights
YOLOv3-608	COCO trainval	test-dev	57.9	140.69 Bn	20	cfg	weights
YOLOv3-tiny	COCO trainval	test-dev	33.1	5.56 Bn	220	cfg	weights
YOLOv3-spp	COCO trainval	test-dev	60.6	141.45 Bn	20	cfg	weights

Fig 4.18: COCO dataset performance

It is crucial to remember that a pre-trained model's performance might not be enough for all use scenarios, particularly if the precise items that need to be recognised are poorly represented in the training dataset. To attain the best performance in these circumstances, fine-tuning the pre-trained model on a particular dataset may be required.

As opposed to building a model from scratch, you may save time and money by employing a pre-trained YOLO model with pre-determined weights and take advantage of the YOLO architecture's outstanding accuracy and performance.

Chapter-5

CONCLUSIONS

5.1 Conclusions

The goal of the platform is to encourage students to engage in sustainable and responsible consumerism by giving them a place to purchase and sell products they no longer need or want. Students may lessen their carbon impact and help create a more sustainable future by recycling items.

The machine learning model of JuitOLX that uses the YOLOv3 algorithm is one of its key benefits. This attribute makes sure everything being offered on the site is secure and lawful, shielding both buyers and sellers from any legal problems. With this business model in place, JuitOLX offers all users a reliable and safe buying and selling experience.

Additionally, the MERN stack, which was used in the development of JuitOLX, gives the website a stable and scalable foundation. The real-time updates function gives buyers and sellers access to immediate transaction updates, resulting in a smooth and open buying and selling process. Buyers may easily locate what they need using the website's search capability, which lets them filter postings depending on their interests.

JuitOLX has the potential to be a role model for ethical and sustainable e-commerce platforms in general. JuitOLX may improve both the lives of students and the environment by fostering sustainability, responsible consumerism, and legal compliance. The platform's future development and growth are interesting to watch, and we can anticipate that it will become more and more crucial to the movement for sustainable consumerism.

5.2 Future Scope

- The use of machine learning has significantly increased the success of eCommerce businesses in addition to the considerable contributions made by social media and the development of cutting-edge apps. On eCommerce websites, artificial intelligence has become a crucial component of the recommendation system for users, helping to make suggestions for products based on users' past purchases, search histories, and other relevant factors. Machine learning algorithms have made it possible for eCommerce businesses to customise the consumer experience, enhancing conversion and customer retention rates.
- Additionally, machine learning has made it possible for eCommerce businesses to analyse customer data and behaviour, enabling them to decide on inventory, pricing, and marketing strategies with confidence. Companies may better fulfil the demands of their target market by customising their product offerings and marketing efforts by analysing data such as client demographics, buying behaviours, and preferences. Additionally, machine learning algorithms can assist eCommerce businesses in streamlining their logistics and supply chain to increase effectiveness and lower costs.

Overall, the use of machine learning technology in eCommerce businesses has transformed the sector and given businesses invaluable knowledge on client preferences and behaviour. Artificial intelligence will probably be used more frequently as eCommerce develops and grows, enabling businesses to remain ahead of the competition and provide customers a customised, effective, and easy purchasing experience.

5.3 Applications Contributions

Numerous potential uses for the juitOLX project exist in the field of ethical and sustainable eCommerce.

- First off, institutions and colleges may utilise the platform to promote ethical and economical buying and selling among students. The platform may assist universities and colleges in being more ecologically friendly and responsible by encouraging the reuse of items and minimising trash.
- The juitOLX platform may also be modified to benefit other communities and organisations, such as local communities or neighbourhoods. The platform may encourage responsible consumption, minimise waste, and aid in the development of more sustainable communities by giving people a place to purchase and sell things they no longer need.
- Additionally, other eCommerce platforms can use the YOLOv3 algorithm to incorporate a machine learning model to guarantee that all products being sold are secure and compliant. eCommerce businesses may safeguard their clients and avert any legal problems by identifying and stopping the sale of unlawful goods.
- Other eCommerce projects can leverage the MERN stack that was used to construct juitOLX since it offers a stable and scalable foundation for handling heavy traffic and transaction loads. Customers may benefit from a seamless and open purchasing and selling experience thanks to the real-time updates function.

References

- [1]S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.
- [2]A. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs], Apr. 2018, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1804.02767>.
- [3]S. Lee, J. Kim, and H. Lee, "Improving Performance of YOLOv3 for Real-time Object Detection in Self-Driving Cars," IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 3, pp. 1181-1191, March 2020. DOI: 10.1109/TITS.2019.2915444.
- [4]S. Gupta and R. Jain, "Real-time Object Detection in Security Camera Systems using YOLOv3," IEEE Transactions on Circuits and Systems for Video Technology, vol. 30, no. 3, pp. 736-746, March 2020. DOI: 10.1109/TCSVT.2019.2914402.
- [5]W.-C. Kang et al., "Object Detection Using Deep Learning: A Review," arXiv:1907.09408 [cs], Jul. 2019, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1907.09408>.
- [6]W.-C. Kang et al., "Object Detection Using Deep Learning: A Review," arXiv:1907.09408 [cs], Jul. 2019, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1907.09408>.
- [7]A. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs], Apr. 2018, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1804.02767>.

[8] A. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs], Apr. 2018, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1804.02767>.

[9]J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," arXiv:1612.08242 [cs], Dec. 2016, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1612.08242>.

[10]J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," arXiv:1612.08242 [cs], Dec. 2016, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1612.08242>.

[11]J. Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection," arXiv:1506.02640 [cs], June. 2015, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1506.02640>.

[12]A. Ahmed and M. Islam, "Multi-object Detection in Surveillance Systems using YOLOv3: Benefits and Privacy Concerns," IEEE Transactions on Information Forensics and Security, vol. 16, pp. 1464-1476, 2021. DOI: 10.1109/TIFS.2021.3071246.

[13]A. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs], Apr. 2018, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1804.02767>.

[14].A. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs], Apr. 2018, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1804.02767>.

[15]J. Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection," arXiv:1506.02640 [cs], June. 2015, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1506.02640>.

[16]A. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs], Apr. 2018, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1804.02767>

[17]A. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs], Apr. 2018, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1804.02767>.

[18] W.-C. Kang et al., "Object Detection Using Deep Learning: A Review," arXiv:1907.09408 [cs], Jul. 2019, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1907.09408>.

[19]A. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs], Apr. 2018, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1804.02767>.

[20]A. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs], Apr. 2018, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1804.02767>.

[21]K. Singh and S. Kumar, "Object Recognition in Autonomous Vehicles using YOLOv3," IEEE Transactions on Intelligent Transportation Systems, vol. 20, no. 5, pp. 1938-1948, May 2019. DOI: 10.1109/TITS.2018.2842407.

[22]S. Choi, S. Lee, and J. Kim, "Real-time Object Detection in Medical Imaging using YOLOv3," IEEE Transactions on Medical Imaging, vol. 39, no. 10, pp. 3289-3299, Oct. 2020. DOI: 10.1109/TMI.2020.2981618.

[23]M. Singh and N. Sharma, "Challenges and Opportunities in Employing YOLOv3 for Object Detection in Medical Imaging: A Review," IEEE Journal

of Biomedical and Health Informatics, vol. 25, no. 5, pp. 1535-1544, Sept. 2021. DOI: 10.1109/JBHI.2020.3028748.

[24]A. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs], Apr. 2018, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1804.02767>.

[25]J. Zhang, X. Zhang, and W. Liu, "Real-time Object Detection for Robotic Applications using YOLOv3," IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 834-841, Apr. 2020. DOI: 10.1109/LRA.2020.2969269.

[26]G. Wu, X. Li, and J. Liu, "Dealing with Occlusions in Object Detection: A Survey," IEEE Transactions on Circuits and Systems for Video Technology, vol. 30, no. 3, pp. 673-686, Mar. 2020. DOI: 10.1109/TCSVT.2019.2908843.

[27]J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," arXiv:1612.08242 [cs], Dec. 2016, Accessed: May 04, 2023. [Online]. Available: <http://arxiv.org/abs/1612.08242>.

[28]M. Li, Y. Wu, and Z. Wang, "A Review of YOLOv3 for Object Detection: Advantages, Challenges, and Applications," IEEE Access, vol. 7, pp. 91993-92005, Jul. 2019. DOI: 10.1109/ACCESS.2019.2925283.

[29]S. Sharma, S. Kumar, and A. Jain, "A Comprehensive Review of YOLOv3 for Object Detection: Advantages, Limitations, and Future Directions," IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 8, pp. 3373-3383, Aug. 2020. DOI: 10.1109/TITS.2019.2967283.

[30] Q.-C. Mao, H.-M. Sun, Y.-B. Liu and R.-S. Jia, "Mini-YOLOv3: Real-Time Object Detector for Embedded Applications," in IEEE Access, vol. 7, pp. 133529-133538, 2019, doi: 10.1109/ACCESS.2019.2941547.

[31] J.Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv preprint arXiv:1804.02767, 2018. [Online]. Available: <https://arxiv.org/abs/1804.02767>. [Accessed: May 8, 2023].

[32] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," arXiv preprint arXiv:1708.02002, 2018. [Online]. Available: <https://arxiv.org/abs/1708.02002>.

[33]C. Chun, L. Dian, Y. Zhi Jiang, J. Wang, and C. Zhang, "YOLOv3: Face Detection in Complex Environments," *International Journal of Computational Intelligence Systems*, vol. 13, no. 1, pp. 1153-1160, 2020, doi: 10.2991/ijcis.d.200805.002.