**MIT WORLD PEACE UNIVERSITY**
**School of Computer Engineering and Technology**
124, Paud Road, Kothrud, Pune 411038, Maharashtra- India
Web: [www.mitwpu.edu.in](http://www.mitwpu.edu.in)

**BE Capstone  Project Report**

# IMAGE UPLOAD WEB APP WITH OBJECT DETECTION

By- Shivam Agarwal

Shubham Jha

Satyajeet Salhunke

Sanjeet Jha

Project Guide :- Prof. Prakash Waghmore

Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

SCHOOL OF COMPUTER ENGINEERING AND TECHNOLOGY

## C E R T I F I C A T E

**This is to certify**

**that,**

**Shivam Agrawal**

**Shubham Jha**

**Satyajeet Salunkhe**

**Sanjeet Jha**

**of B.E.(Computer Engineering and Technology) have partially completed their project report on Image Upload Web App with Object Detection and have submitted this End term partial report towards fulfillment of the requirement for the Degree - Bachelor of Computer Science & Engineering (B.Tech-CSE) for the academic year 2020-2021.**

| | |
|---|---|
| **[Dr/ Prof. Prakash Waghmore]** | **[Dr. Mangesh Bedekar]** |
| **Project Guid** | **Program Head** |
| **School of CET** | **School of CET** |
| **MIT World Peace University, Pune** | **MIT World Peace University, Pune** |

**Date:**

# ACKNOWLEDGEMENT

We would like to thank **MIT World Peace University** for giving Us this wonderful opportunity for making this project.
Also we would like to express our gratitude to the School of CET for allowing us to choose our own BE project topic and form our own groups.
Next we would like to thank our project supervisor Prof. Prakash Waghmore for always being in touch with us and Guiding us  and for arranging presentation meetings.
Finally we thank all the faculties involved and making project presentations possible in virtual mode.

# INDEX

# Introduction

More than 90% of adults now have mobile phones most of which allow them to take photos and instantly store and share them on online platforms. Half of all Internet users (54%) share original photos and videos online, and an increasing number are using specific photo storing applications
User studies have started to explore what type of content is being shared, categories of users, and the role of social features within a photo-based social network .
Image upload webapp is a photo sharing , object detection and classification and storage service.
The service will be free and unlimited. The service automatically analyzes photos, identifying various visual features and classes of objects.
users can search for a photo, with the service returning results from 3 major categories. People, Places and Things.

The computer vision of photos recognizes classes of objects ,grouping similar one together and subject matters including buildings, animals, food and more.
Humans glance at an image and instantly know what objects are in the image, where they are, and how they interact. The human visual system is fast and accurate, allowing us to perform complex tasks . Fast, accurate algorithms for object detection would allow computers to drive cars without specialized sensors, enable assistive devices to convey real-time scene information to human users, and unlock the potential for general purpose, responsive robotic systems.

We reframe object detection as a single regression problem, straight from image pixels to bounding box coordinates and class probabilities. Using the  system, you only look once (YOLO) at an image to predict what objects are present and where they are. YOLO is refreshingly simple. A single convolutional network simultaneously predicts multiple bounding boxes and class probabilities for those boxes. YOLO trains on full images and directly optimizes detection performance. This unified model has several benefits over traditional methods of object detection.

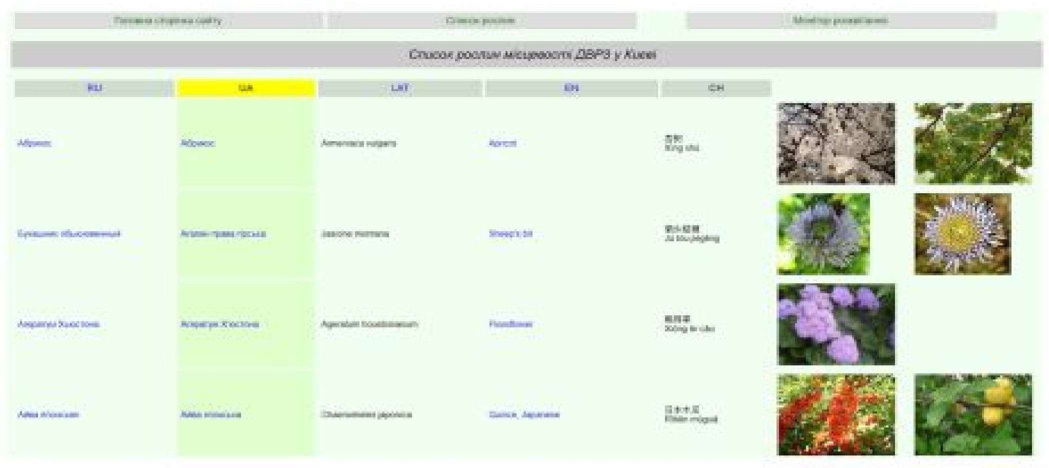
YOLO is extremely fast. Since we frame detection as a regression problem we don't need a complex pipeline. We simply run our neural network on a new image at test time to predict detections. The base network runs at 45 frames per second with no batch processing on a Titan X GPU and a fast version runs at more than 150 fps. This means we can process streaming video in real-time with less than 25 milliseconds of latency. Furthermore, YOLO achieves more than twice the mean average precision of other real-time systems.

YOLO predicts multiple bounding boxes per grid cell. At training time we only want one bounding box predictor to be responsible for each object. We assign one predictor to be "responsible" for predicting an object based on which prediction has the highest current IOU with the ground truth. This leads to specialization between the bounding box predictors. Each predictor gets better at predicting certain sizes, aspect ratios, or classes of object, improving overall recall.

# Literature Survey

**Model experiment**

To provide an experiment and compare results with keys for each plant, 500 photos from online-classifier “The list of plants of the Dneprovskiy district of Kiev” (Fig. 2) were taken. The online-classifier contains the pictures of each kind of the plants and its determination names. Photos were characterized by the method described in 3.2 due to the different quality of the photos and collected.



Головна сторінка сайту | Список рослин | [illegible]

Список рослин місцевості ДВРЗ у Києві

| RU | UA | LAT | EN | CH |
|---|---|---|---|---|
| Абрикос | Абрикос | Armeniaca vulgaris | Apricot | 杏树 Xìng shù |
| Букашник обыкновенный | [illegible] | Jasione montana | Sheep's bit | [illegible] |
| Агератум Хьюстона | Агератум Хюстона | Ageratum houstonianum | Flossflower | [illegible] |
| Айва японская | Айва японська | Chaenomeles japonica | Quince, Japanese | 日本木瓜 Rìběn mùguā |

**The general method of photo analysis**

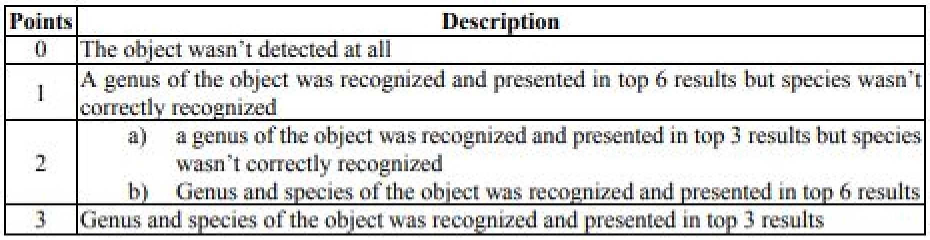
Photo’s quality is an important factor to Google Photos. Therefore, it is necessary to classify each photo by main quality components – composition, resolution, digital noise. Main photos quality criteria are presented in table .

| Quality | Analyzed object`s resolution, Mpx | Gray noise | Color noise | Analyzing object |
|---|---|---|---|---|
| Bad | <0.3 | High | High | Not clearly visible |
| Middle | 0.3–3 | Middle | Middle | Clearly visible |
| Good | >3 | Low | Low | Perfectly visible |

## Data collection and analysis

To collect data, we developed the database with front-end and back-end development. Each photo was classified by the image quality using the method described , and its characteristics such as type (tree, bush, grass) and presented part of the plant (flower, leaf, stem, fruit). The mark of the analyzing process was inputted too.

The output interface looked like a table to provide the visualization and dynamic of the research process. Google proposed a few results of the analysis to the user. Therefore, the results were classified on 0, 1, 2 or 3 points. Sometimes cropping of the photos was used, in this case, one point was deducted.



| Points | Description |
|---|---|
| 0 | The object wasn't detected at all |
| 1 | A genus of the object was recognized and presented in top 6 results but species wasn't correctly recognized |
| 2 | a) a genus of the object was recognized and presented in top 3 results but species wasn't correctly recognized<br>b) Genus and species of the object was recognized and presented in top 6 results |
| 3 | Genus and species of the object was recognized and presented in top 3 results |

Results were collected on the database. To provide an analysis of the requests to a database prepared and provided. The requests were prepared to take into account the aims of the work. To process the results MS Excel 365 was used.

## Search by image to reveal copies of a known image

A more recent type of searching, which deals also mainly with images, is search by image or "reverse image search" as it is named by the leading search system that is provided by Google. In this type of searching, a query consists not of a word or a combination of words, but of one image; this image is of course already known by the user who hopes to find images that are relevant in the context of the user's information need. The search system can retrieve mainly other images with similar distributions of picture elements. Therefore such a search can reveal mainly so-called "copies" of the image that was used in the query.

This method can be quite useful to detect plagiarism. More generally, this type of search allows us to determine reuse of an image that has been available / shared / published on the Internet; this is welcomed for instance by managers of digital libraries that include images (see for instance [5, 9]). Finding images that are semantically similar may be thought of as another obvious application, but this is hindered by the so-called semantic gap, as explained further

## Evaluation of the automatic classifications in Albums

The quality of the automatic classifications / annotations / tags was evaluated as follows. For each Album with corresponding simple name / annotation / tag, that has been automatically created by Google, each image that has been classified in that category / class was inspected; then it was evaluated if the classification made sense, i.e. was correct or not. In other words, a bimodal model was used. This is a simple, rudimentary model as the quality of a classification has many aspects / dimensions and can be assigned many levels between correct or not, 1 or 0. This approach is justified in view of the modest aims and size of this case study.

## Object Detection:

Object detection is a computer vision technique for locating instances of objects in images or videos. Object detection algorithms typically leverage machine learning or deep learning to produce meaningful results. When humans look at images or video, we can recognize and locate objects of interest within a matter of moments. The goal of object detection is to replicate this intelligence using a computer.Object detection is a key technology behind advanced driver assistance systems (ADAS) that enable cars to detect driving lanes or perform pedestrian detection to improve road safety. Object detection is also useful in applications such as video surveillance or image retrieval systems.

# APPROACH

**(For Object Detection)**

## What is Object Recognition?

Object recognition is a general term to describe a collection of related computer vision tasks that involve identifying objects in digital photographs.

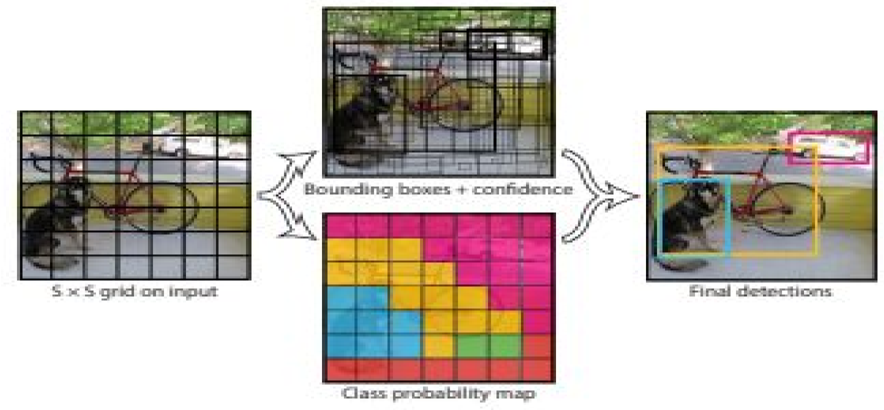
Image classification involves predicting the class of one object in an image. Object localization refers to identifying the location of one or more objects in an image and drawing a bounding box around their extent. Object detection combines these two tasks and localizes and classifies one or more objects in an image.

When a user or practitioner refers to “object recognition“, they often mean “object detection“.

Image Classification: Predict the type or class of an object in an image.
Input: An image with a single object, such as a photograph.
Output: A class label (e.g. one or more integers that are mapped to class labels).
Object Localization: Locate the presence of objects in an image and indicate their location with a bounding box.
Input: An image with one or more objects, such as a photograph.
Output: One or more bounding boxes (e.g. defined by a point, width, and height).
Object Detection: Locate the presence of objects with a bounding box and types or classes of the located objects in an image.
Input: An image with one or more objects, such as a photograph.
Output: One or more bounding boxes (e.g. defined by a point, width, and height), and a class label for each bounding box.

One further extension to this breakdown of computer vision tasks is object segmentation, also called “object instance segmentation” or “semantic segmentation,” where instances of recognized objects are indicated by highlighting the specific pixels of the object instead of a coarse bounding box.

From this breakdown, we can see that object recognition refers to a suite of challenging computer vision tasks.



Overview of Object Recognition Computer Vision Tasks
Most of the recent innovations in image recognition problems have come as part of participation in the ILSVRC tasks.
This is an annual academic competition with a separate challenge for each of these three problem types, with the intent of fostering independent and separate improvements at each level that can be leveraged more broadly. For example, see the list of the three corresponding task types below taken from the 2015 ILSVRC review paper:

Image classification: Algorithms produce a list of object categories present in the image.

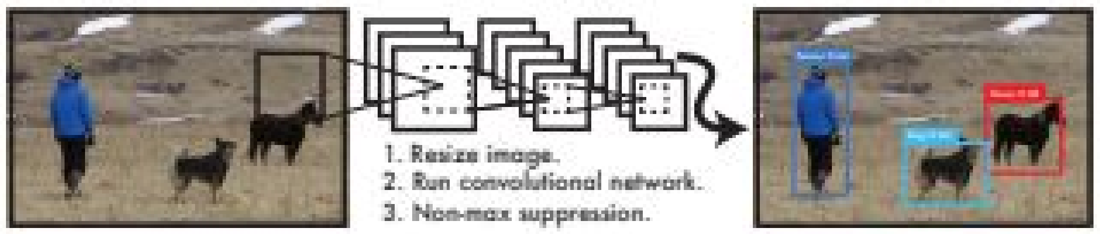
Single-object localization: Algorithms produce a list of object categories present in the image, along with an axis-aligned bounding box indicating the position and scale of one instance of each object category.

Object detection: Algorithms produce a list of object categories present in the image along with an axis-aligned bounding box indicating the position and scale of every instance of each object category.

We can see that “Single-object localization” is a simpler version of the more broadly defined “Object Localization,” constraining the localization tasks to objects of one type within an image, which we may assume is an easier task.

Below is an example comparing single object localization and object detection, taken from the ILSVRC paper. Note the difference in ground truth expectations in each case.
Comparison Between Single Object Localization and Object Detection.

The performance of a model for image classification is evaluated using the mean classification error across the predicted class labels. The performance of a model for single-object localization is evaluated using the distance between the expected and predicted bounding box for the expected class. Whereas the performance of a model for object recognition is evaluated using the precision and recall across each of the best matching bounding boxes for the known objects in the image.



Now that we are familiar with the problem of object localization and detection, let's take a look at some recent top-performing deep learning models.
R-CNN Model Family

The R-CNN family of methods refers to the R-CNN, which may stand for "Regions with CNN Features" or "Region-Based Convolutional Neural Network," developed by Ross Girshick, et al.

This includes the techniques R-CNN, Fast R-CNN, and Faster-RCNN designed and demonstrated for object localization and object recognition.

Let's take a closer look at the highlights of each of these techniques in turn.
R-CNN

The R-CNN was described in the 2014 paper by Ross Girshick, et al. from UC Berkeley titled "Rich feature hierarchies for accurate object detection and semantic segmentation."

It may have been one of the first large and successful application of convolutional neural networks to the problem of object localization, detection, and segmentation. The approach was demonstrated on benchmark datasets, achieving then state-of-the-art results on the VOC-2012 dataset and the 200-class ILSVRC-2013 object detection dataset.

Their proposed R-CNN model is comprised of three modules; they are:

Module 1: Region Proposal. Generate and extract category independent region proposals, e.g. candidate bounding boxes.
Module 2: Feature Extractor. Extract feature from each candidate region, e.g. using a deep convolutional neural network.
Module 3: Classifier. Classify features as one of the known class, e.g. linear SVM classifier model.

The architecture of the model is summarized in the image below, taken from the paper.
Summary of the R-CNN Model Architecture

Summary of the R-CNN Model ArchitectureTaken from Rich feature hierarchies for accurate object detection and semantic segmentation.

A computer vision technique is used to propose candidate regions or bounding boxes of potential objects in the image called "selective search," although the flexibility of the design allows other region proposal algorithms to be used.

The feature extractor used by the model was the AlexNet deep CNN that won the ILSVRC-2012 image classification competition. The output of the CNN was a 4,096 element vector that describes the contents of the image that is fed to a linear SVM for classification, specifically one SVM is trained for each known class.

It is a relatively simple and straightforward application of CNNs to the problem of object localization and recognition. A downside of the approach is that it is slow, requiring a CNN-based feature extraction pass on each of the candidate regions generated by the region proposal algorithm. This is a problem as the paper describes the model operating upon approximately 2,000 proposed regions per image at test-time.

Python (Caffe) and MatLab source code for R-CNN as described in the paper was made available in the R-CNN GitHub repository.
Fast R-CNN

Given the great success of R-CNN, Ross Girshick, then at Microsoft Research, proposed an extension to address the speed issues of R-CNN in a 2015 paper titled "Fast R-CNN."

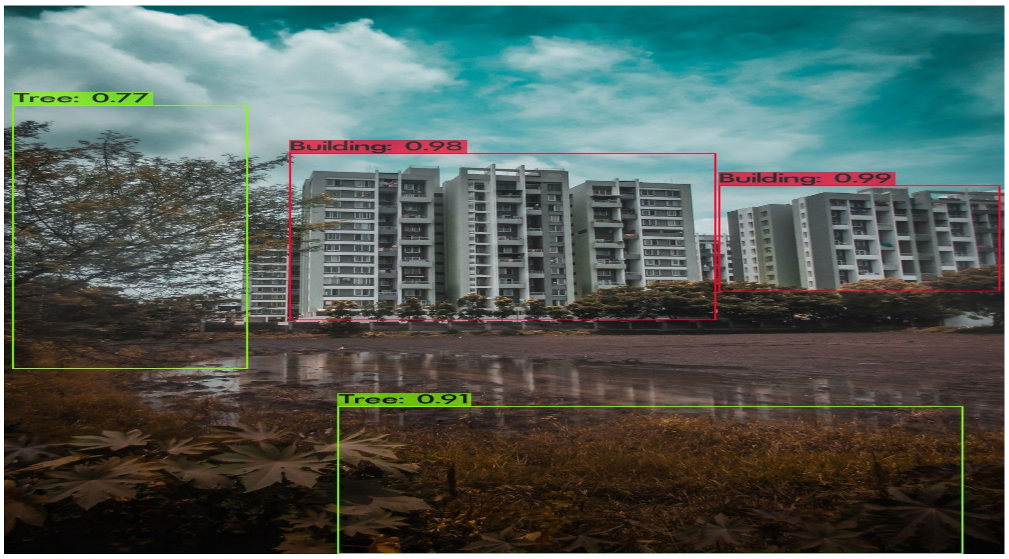
The paper opens with a review of the limitations of R-CNN, which can be summarized as follows: Training is a multi-stage pipeline. Involves the preparation and operation of three separate models.

Training is expensive in space and time. Training a deep CNN on so many region proposals per image is very slow.
Object detection is slow. Make predictions using a deep CNN on so many region proposals is very slow.

A prior work was proposed to speed up the technique called spatial pyramid pooling networks, or SPPnets, in the 2014 paper “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition.” This did speed up the extraction of features, but essentially used a type of forward pass caching algorithm.

Fast R-CNN is proposed as a single model instead of a pipeline to learn and output regions and classifications directly.

The architecture of the model takes the photograph a set of region proposals as input that are passed through a deep convolutional neural network. A pre-trained CNN, such as a VGG-16, is used for feature extraction. The end of the deep CNN is a custom layer called a Region of Interest Pooling Layer, or RoI Pooling, that extracts features specific for a given input candidate region.



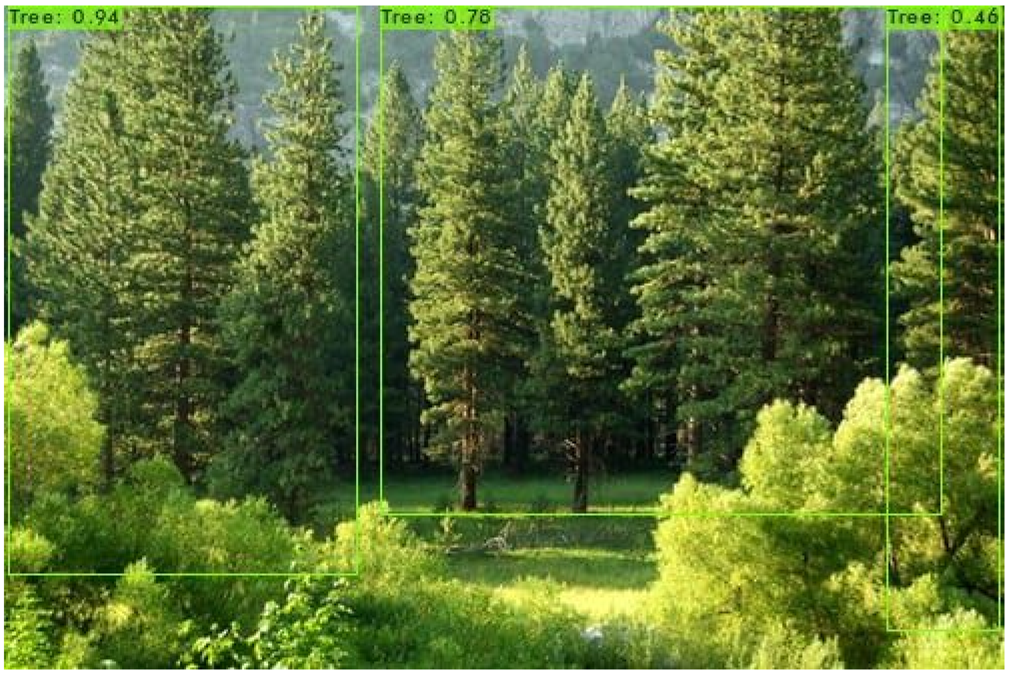
[**Fig B**: Prediction output for the building and the tree class]

The output of the CNN is then interpreted by a fully connected layer then the model bifurcates into two outputs, one for the class prediction via a softmax layer, and another with a linear output for the bounding box. This process is then repeated multiple times for each region of interest in a given image.

The architecture of the model is summarized in the image below, taken from the paper.
Summary of the Fast R-CNN Model Architecture

Summary of the Fast R-CNN Model Architecture.
Taken from: Fast R-CNN.

The model is significantly faster to train and to make predictions, yet still requires a set of candidate regions to be proposed along with each input image.



[Fig C. Prediction output for tree class]

# Problem Statement

**Project Scope**

a. This website provides an image upload Web Service which detects classes within images like if its a tree and automatically group them into albums.
b. It has a search engine which shows images of the text query.
c. The App deployment is automated using Kubernetes

**Project Assumptions**

Made regarding budgeting , scheduling constraints, and resource . Assumptions made about these factors are circumstances that are presumed to be true in the future, and project strategies are built around them.

I. Resource Assumption
II. Datasets will be taken from Kaggle or compiled using a python script
III. Next we will use these dataset to train our model

**Technology assumption**

We are using YOLOv3 object detection algorithm for training our dataset and detect the objects in the image.

**Cost assumption.**

All the softwares we are using is Open Source. The only cost that can arise is due to AmazonAws Hosting on Ec2 and Kubernetes Engine.

1. **Time-based assumptions**

   We have divided the Project in 3-3 months,first 3 months we will do project planning & designing and next 3 months will do project implementation.

**Project Limitation:-**

We are not able to find all the datasets on Kaggle and other sources, for that we will need to use a script which downloads images from Google images. Since it's not labelled we will need to manually create labels which is a very lengthy process.

**Project Objective:-**

It is Web App which automatically detects objects in an image and creates an album for the same.

# .Project Requirements

1. **Resources**

    1. **Human Resources**
        1.1. 4 persons(Team Members)

    2. **Reusable Software Components** :- NA

    3. **Software requirements**
        3.1. S/W:-
        3.2. Google Colab
        3.3. VS code
        3.4. NodeJS
        3.5. ReactJS
        3.6. Django (Python)
        3.7. YOLOv3

    4. **H/W requirements**
        4.1. Any Operating System like Ubuntu
        4.2. 4 GB Ram
        4.3. 2GHZ processor

2. **Requirements Rationale**

| **Requirements** | **Rationale** |
|---|---|
| Functional Requirements | System tasks functions such as taking image input and then processing it into detection |
| Performance Requirements | The Objected Detected should have a high accuracy |
| Usability Requirements | The frontend will be very user-friendly |
| Interface Requirements | Website |
| Operational Requirements | Datasets will be compiled manually and from Kaggle |
| Design Constraints | System is complex but yet our design while taking considerations of different aspects is simple. |
| Cost and Schedule Constraints | All software are open source,but amazon aws hosting there can be charges for extra resources used. |

[Table 1.Requirement Rationale]

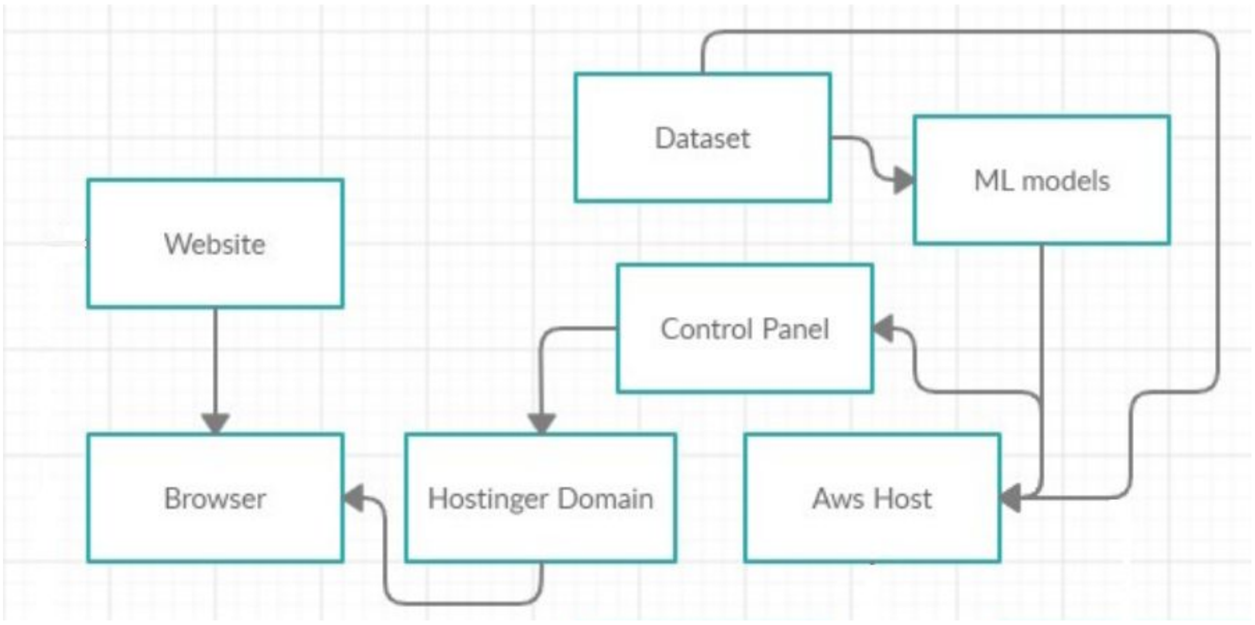
## 3. Risk Management

### 3.1. Project Risk factors in Table format

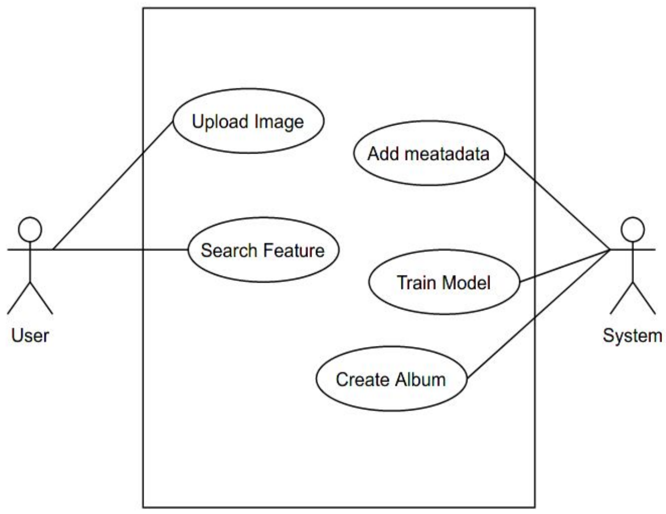
| **Low Risk** | **Medium Risk** | **High Risk** |
|---|---|---|
| Scalability is not implemented properly | Object Detection can't be precise and not detect any objects or return incorrect results | Albums are not being created automatically |

Table 2. Project Risk

## 4. System Analysis Proposed Architecture/ high level design of the project



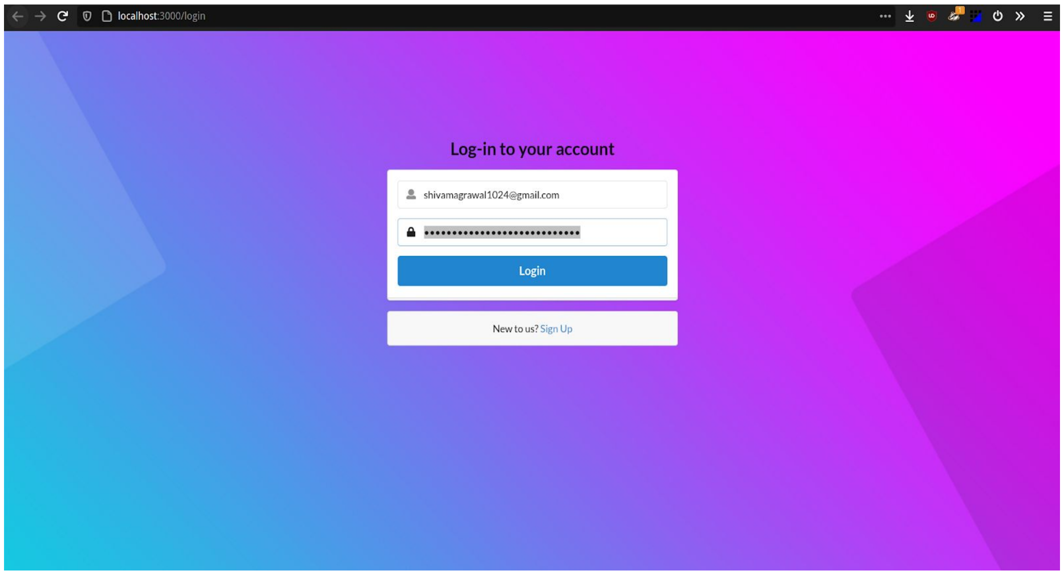
[Fig 2. Website Architecture ]



[Fig 3. Use Case Diagram]

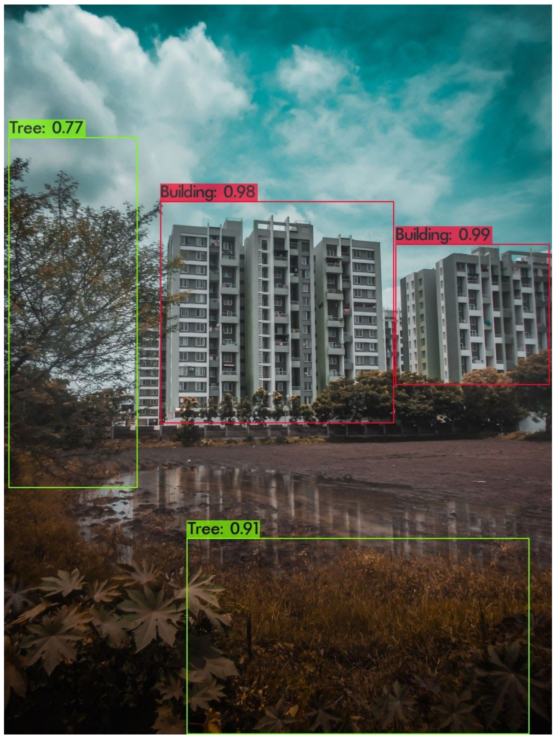
## Results

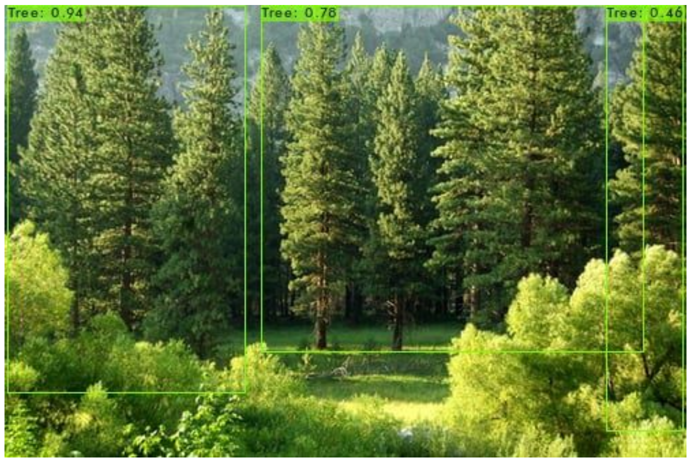
We created a Login page layout on which the user has to fill login credentials in order to log in to their account. We are thinking of adding captcha at the end for robot verification. We are trying different React.js templates for login as well as Dashboard section.



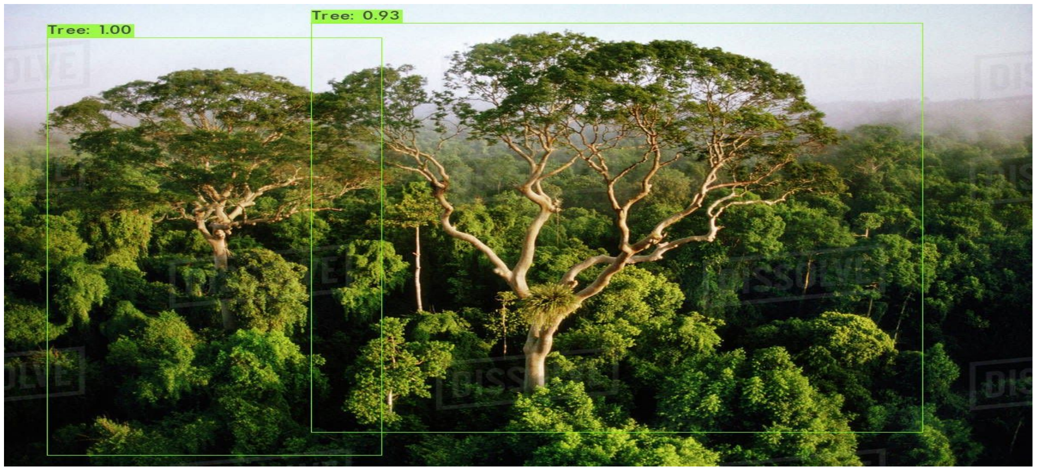
We are also working on Creating a user-friendly Dashboard with features like uploading Photos and displaying already uploaded pictures in the form of albums under different sections.

We have created a Data set for classes like( trees, buildings , birds, Airplanes..) and labeled them in YOLO format.
We used Google colab to first load the dataset , divide them into Test set and Train set and used yolo object detection to train the dataset.





As we can see after training the dataset on YOLO the model is able to detect classes of objects in the image which were previously undetectable.
Yolo is able to detect the images accurately upto 98% in some cases if the image is clearly detected. So we are thinking of creating datasets on more such classes and then train them.
Then we are going to group them according to the classes that we have created.



## Research Gap

The first size challenge is that for users with large photo collections there is simply too much metadata. Sending even minimal information (the photo urls, width, height, and timestamps) is many megabytes of data for a full collection, this would directly run counter to our goal of near-instant loading.

The second size challenge is the photos themselves. With modern HDPI screens, even a small photo thumbnail is often 50KB or more. A thousand thumbnails may be 50 megabytes, and not only is that a lot to download, if you try to place them all in the webpage immediately you can slow down the browser. The old Google+ Photos would become sluggish after scrolling through 1000–2000 photos and Chrome would eventually crash the tab after loading 10000.

Scrubbable Photos — the ability to quickly jump to any part of the photo library.
Justified Layout — fill the width of the browser and preserve the aspect-ratio of each photo (no square crops).
60fps Scrolling — ensuring the page remains responsive even when looking at many thousands of photos.
Instantaneous Feel —minimize the time waiting for anything to load.

## Project Plan

- First of all we researched the problem statement and did an in depth analysis of the topic.
- We read Different research papers of google photos ,google lens , numerous case studies of the photo sharing and storing webapps and different algorithms.
- We did research on YOLO and how to train our model by using a pre-trained dataset and then check the results.
- We have to take care of both front end and back end working simultaneously.
- So we will be creating a Login Page in the beginning  using React.js templates.
- Users can enter their credentials and then login into their account.
- Then we are creating a User friendly dashboard on which user can view the uploaded photos and also upload new photos.
- The photos that user uploads will be efficiently managed by NextCloud API.The images will be end to end Encrypted. The idea here is for the user to have its own private cloud which will be managed by the user himself with no interference from 3rd parties.
- For the application deployment it will be done using Docker.
- For the classification of the photos and the image class identification We will train the model using YOLOv3 object detection method, based on Different classes of dataset(people, buildings, trees, pets and more).

| | |
|---|---|
| Planning | 1 Sep 2020 - 27 Sep 2020 |
| Analysis | 27 Sep 2020 - 20 Nov 2020 |
| Implementation | 20 Nov 2021 - April 2021 |
| Deployment | April 2021 |

## <u>Conclusion</u>

Thus, We have found a very accurate and efficient way to train our dataset for object detection and create a model that can accurately detect the classes of the objects in an image. We will be using the Django python framework in the Backend for doing all the tasks and also to connect our YOLO model to the webapp.
For the frontend part we are already using React.Js templates for web app creation and also for creating the dashboard.

# <u>References</u>

- P. Nieuwenhuysen, "Information Discovery and Images A Case Study of Google Photos," 2018 5th International Symposium on Emerging Trends and

Technologies in Libraries and Information Services (ETTLIS), Noida, 2018, pp. 16-21, doi: 10.1109/ETTLIS.2018.8485238.

- K. Chaudhury, S. DiVerdi and S. Ioffe, "Auto-rectification of user photos," 2014 IEEE International Conference on Image Processing (ICIP), Paris, 2014, pp. 3479-3483, doi: 10.1109/ICIP.2014.7025706.

- Lee J. Algorithmic Uses of Cybernetic Memory: Google Photos and a Genealogy of Algorithmically Generated "Memory." *Social Media + Society*. October 2020. doi

- Ferrari, V., Jurie, F. & Schmid, C. From Images to Shape Models for Object Detection. *Int J Comput Vis* **87,** 284–303 (2010).

- A. Mohan, C. Papageorgiou and T. Poggio, "Example-based object detection in images by components," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 4, pp. 349-361, April 2001, doi: 10.1109/34.917571.

- Gupta S., Girshick R., Arbeláez P., Malik J. (2014) Learning Rich Features from RGB-D Images for Object Detection and Segmentation. In: Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8695. Springer, Cham. https://doi.org/10.1007/978-3-319-10584-0_23

- G. Stockman, S. Kopstein and S. Benett, "Matching Images to Models for Registration and Object Detection via Clustering," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-4, no. 3, pp. 229-241, May 1982, doi: 10.1109/TPAMI.1982.4767240.

- R. Huang, J. Pedoeem and C. Chen, "YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers," 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 2018, pp. 2503-2510, doi: 10.1109/BigData.2018.8621865.