**AI-based object counting system for mobile and web deployment**

**Software requirements specification for the AI-based system.**

***Preface***

This SRS is intended for the mobile and web development teams so that they can evaluate the potential system usage. The first SRS version (v1.1) corresponds to the week 1 of the beginning of the project, which is based on the obtaining of requirements process to the software development.

***Introduction***

This software, which is an AI system, will be made with the objective of counting and detecting a type of roses, specifically explorer roses, that are in a given image, with an accuracy greater than 80%. In order to do this, TensorFlow Lite will be used, where files corresponding to the weights and architecture of the model will be exported. Then, it will be used by other systems, specifically by a web platform and a mobile application.

***Glossary***

| AI | Artificial Intelligence |
| --- | --- |
| Batch | Arrange in sets or groups. |
| GPU | Graphical Processing Unit |
| SRS | Software Requirement Specification |
| Neural Network | It is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. |
| Tensor Flow Lite | It Is an open source deep learning framework for on-device inference. |

Table 1. Glossary

***User and System Requirements***

*User requirements definition*

The counting system shall take a rose/s picture/s and count the quantity of roses present in the image. The system can take pictures like: one rose, o a set of flowers.

*System requirements specification*

- The pictures should be uploaded in a png format.

- The system shall output a scalar number which is the number of roses.

- If the system fails for inference (not a good image, not an image of roses) , it should return 0 for the counter and the same images without bounding boxes.

***Functional and non-functional requirements.***

*Functional requirements*

- The input image of the AI-based system should be in any resolution.

- The AI-based system detects roses on the input image.

- The AI-based system counts how many roses are present on the input image.

- The AI-based system returns an output image with a bounding box over roses present on the image and how many roses are present.

*Non-functional requirements*

- Training time should not pass 1 hour.

- The AI-based system should be able to train in 1 GPU.

- The AI-based system should have an accuracy greater than 80%.

- The AI-based system must be optimized for on-device detection.

- The AI-based system should be multiple platform support (Android, iOS)

- The AI-based system must be diverse language programming support (Java,

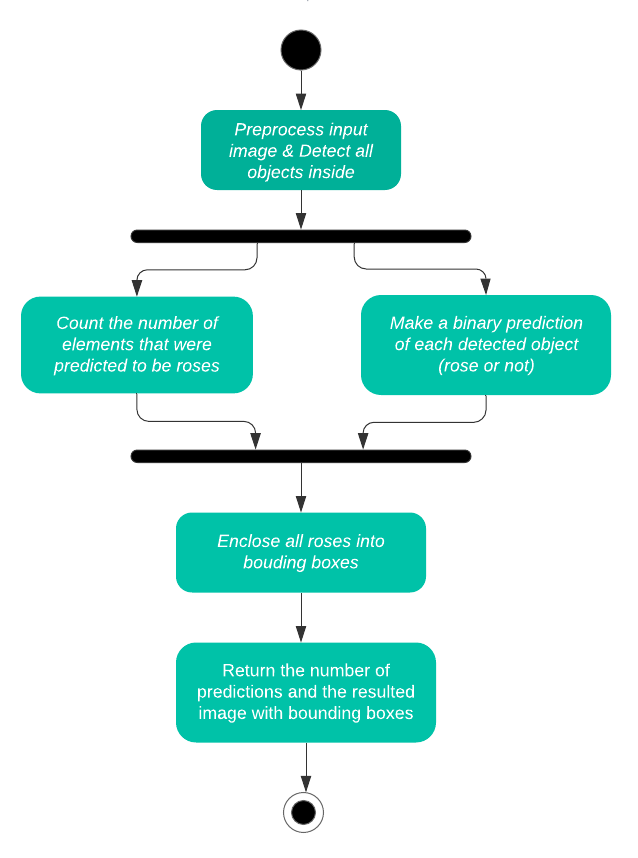
Python)

***Form-based structured requirements specification***

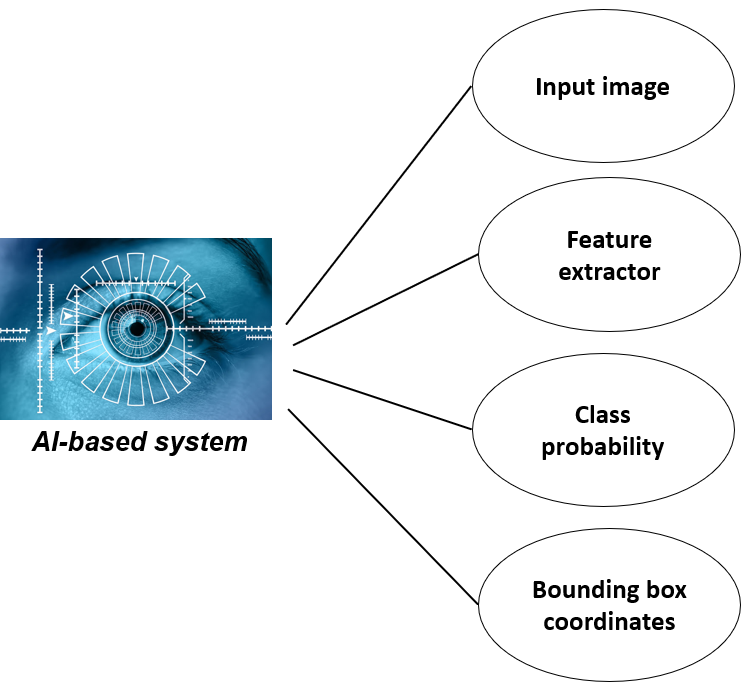
| Function | AI-based object counting |
| --- | --- |
| Description | AI-based system that can detect and count one type of rose in an image with an accuracy greater than 80%. |
| Inputs | Dataset of images of which the user wishes to detect and count a respective category. |
| Source | Images from an image gallery, cameras or through file uploads from a mobile device or computer. |
| Outputs | The respective weights of each node of the neural network and the structure of the model will be saved in a file with a .tflite format. |
| Destination | The files obtained in the model will be sent to the Mobile and Web platforms. |
| Action | AI-based system will work of the following way:   * Enclosing the respective elements of the category within bounding boxes. * Carry out the necessary computation to verify that an element belongs to the specified category. * Showing the elements of the category, whose predicted probability is high. |
| Requirements | Prediction:   * Weights of the previously trained model. * Architecture of that model.   Training:   * Configuration of the training process. |
| Pre-Condition: | * Predicted probability limit higher than 50% to be considered in the counting process. * Dataset for training should be in Pascal VOC format. * Images for training should be in .jpg format. |
| Post-Condition: | Model state updating for each batch of images that pass through the training stage. |
| Side effects | None |

Table 2. Form-based structured requirements specification

***Activity Diagram***



***‘ AI-based system.’ use-case model***



*Input image:*

| **Actors** | Object detection Ai-Based system |
| --- | --- |
| **Description** | The input image is an image with one or more objects and must be in a format that is accepted by the model so the information can be processed by the AI-Based system. |
| **Data** | Data set of full-color red roses images. |
| **Stimulus** | The image comes from a mobile user or web user |
| **Response** | None |
| **Comments** | If the input image is accepted can pass to the feature extractor. |

Table 3. Tabular description of the “Input Image” use-case

*Feature extractor:*

| **Actors** | Object detection AI-Based system |
| --- | --- |
| **Description** | Once the input image is accepted, the AI-based system starts mapping the image pixels into the feature space. |
| **Data** | An array representation of a full color image |
| **Stimulus** | Start when the image is accepted by AI-Based system |
| **Response** | A feature vector which is used to recognize objects (roses) |
| **Comments** | None |

Table 4. Tabular description of the “Feature Extractor” use-case

*Class probability:*

| **Actors** | Object detection Ai-Based system |
| --- | --- |
| **Description** | Fit the classifier to obtain a class probability estimate of a feature vector |
| **Data** | A feature vectors. |
| **Stimulus** | Feature vectors from the feature extractor |
| **Response** | The class probabilities map |
| **Comments** | None |

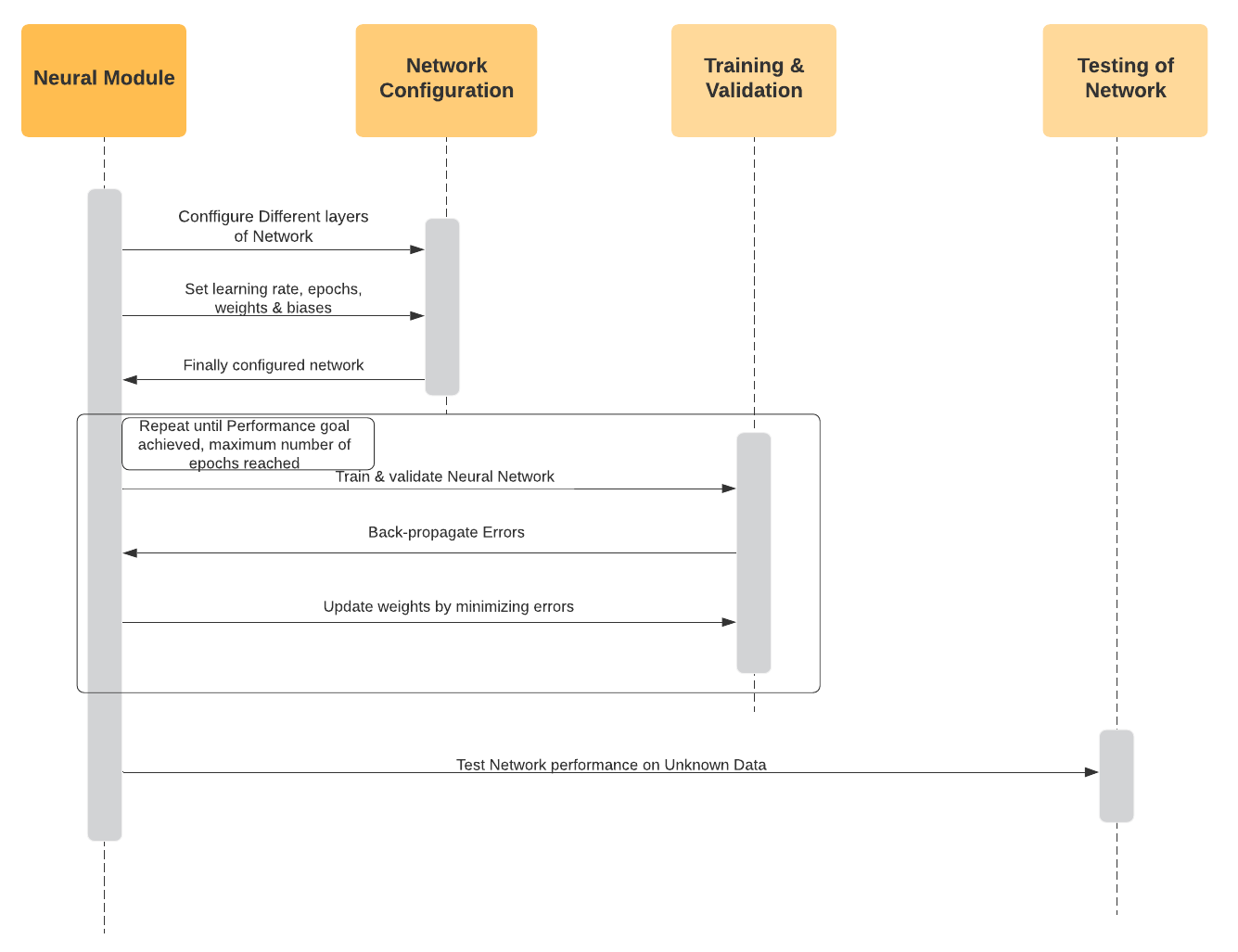
Table 5. Tabular description of the “Class Probability” use-case

*Bounding box coordinate:*

| **Actors** | Object detection Ai-Based system |
| --- | --- |
| **Description** | Locate the presence of roses on the image with a bounding box |
| **Data** | A feature vectors. |
| **Stimulus** | Feature vectors from the feature extractor |
| **Response** | One or more bounding boxes defined by a point, width, and height |
| **Comments** | None. |

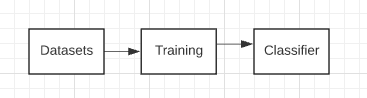
Table 6. Tabular description of the “*Bounding box coordinate*” use-case

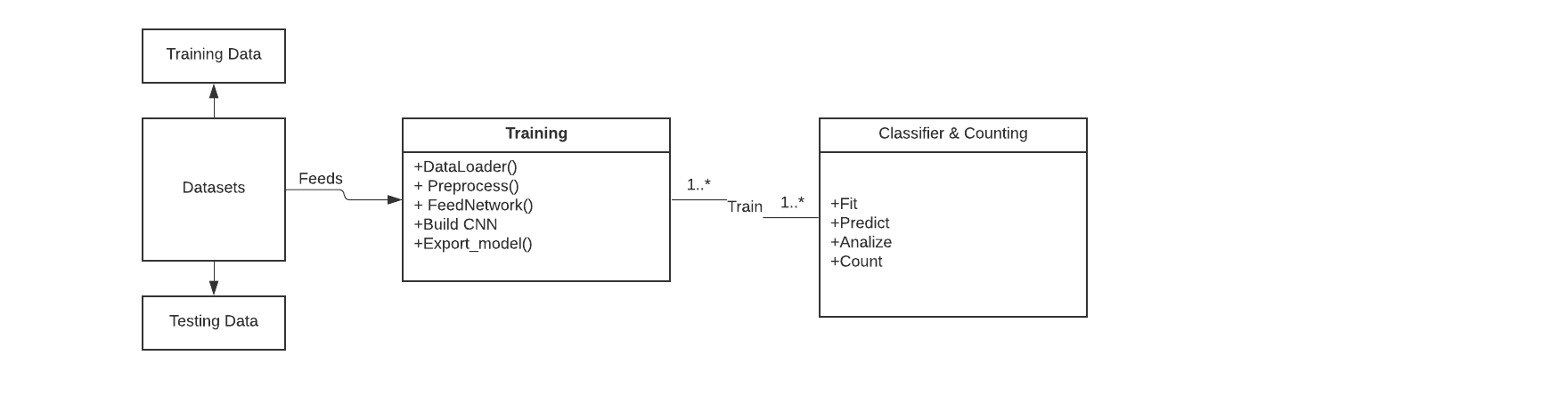
***Sequence diagram***



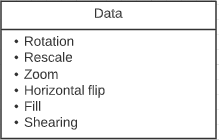
***Class Model***

**a)**

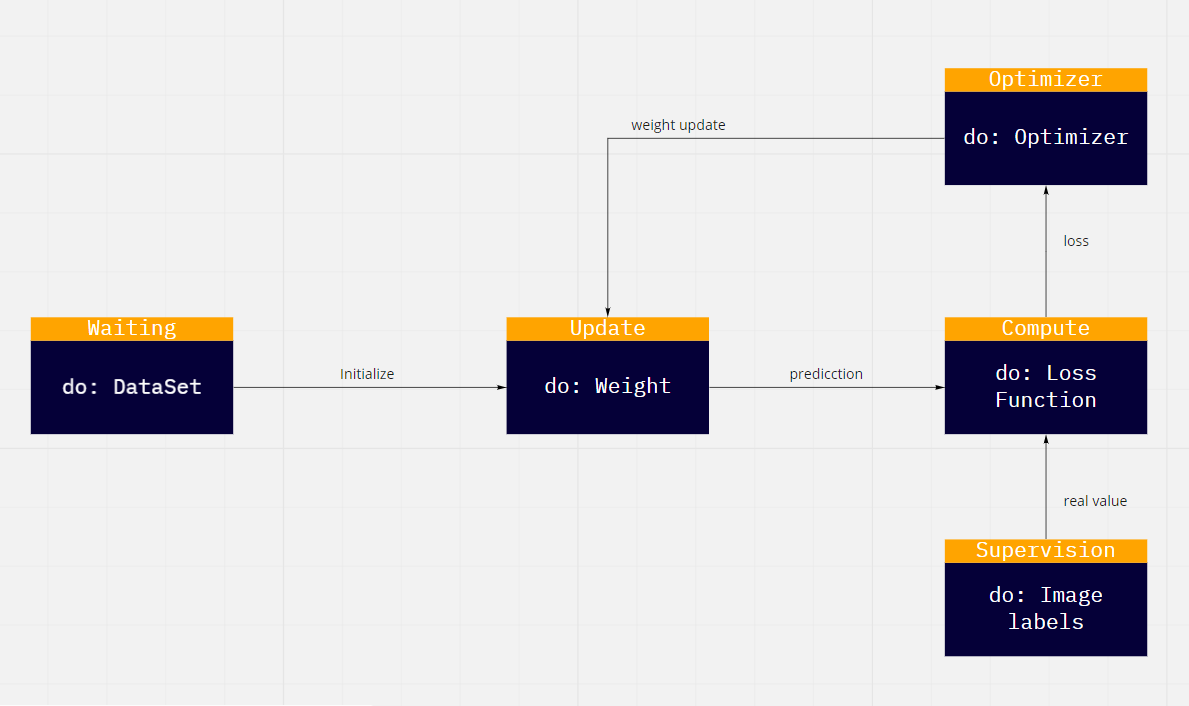
****

**b) **

**c)**

****

***State Model***

****

| **State** | **Description** |
| --- | --- |
| Waiting | Wait for a dataset |
| Update | update the values for each epoch |
| Compute | performs the loss function |
| Optimizer | adjustment in the value of the weights |
| Supervision | respective labels of the image to train |

Table 7. States and their descriptions of the State Model

| **Stimulus** | **Description** |
| --- | --- |
| Initialize | the values of the weights assigned in a random way |
| Prediction | the calculation of how far we are from the real value |
| Loss | the loss in each epoch, value obtained from the loss function |
| Real Value | real values from supervised learning |
| Weight Update | update of the weights in each epoch |

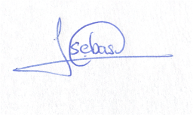
Table 8. Stimulus and descriptions of the State Model

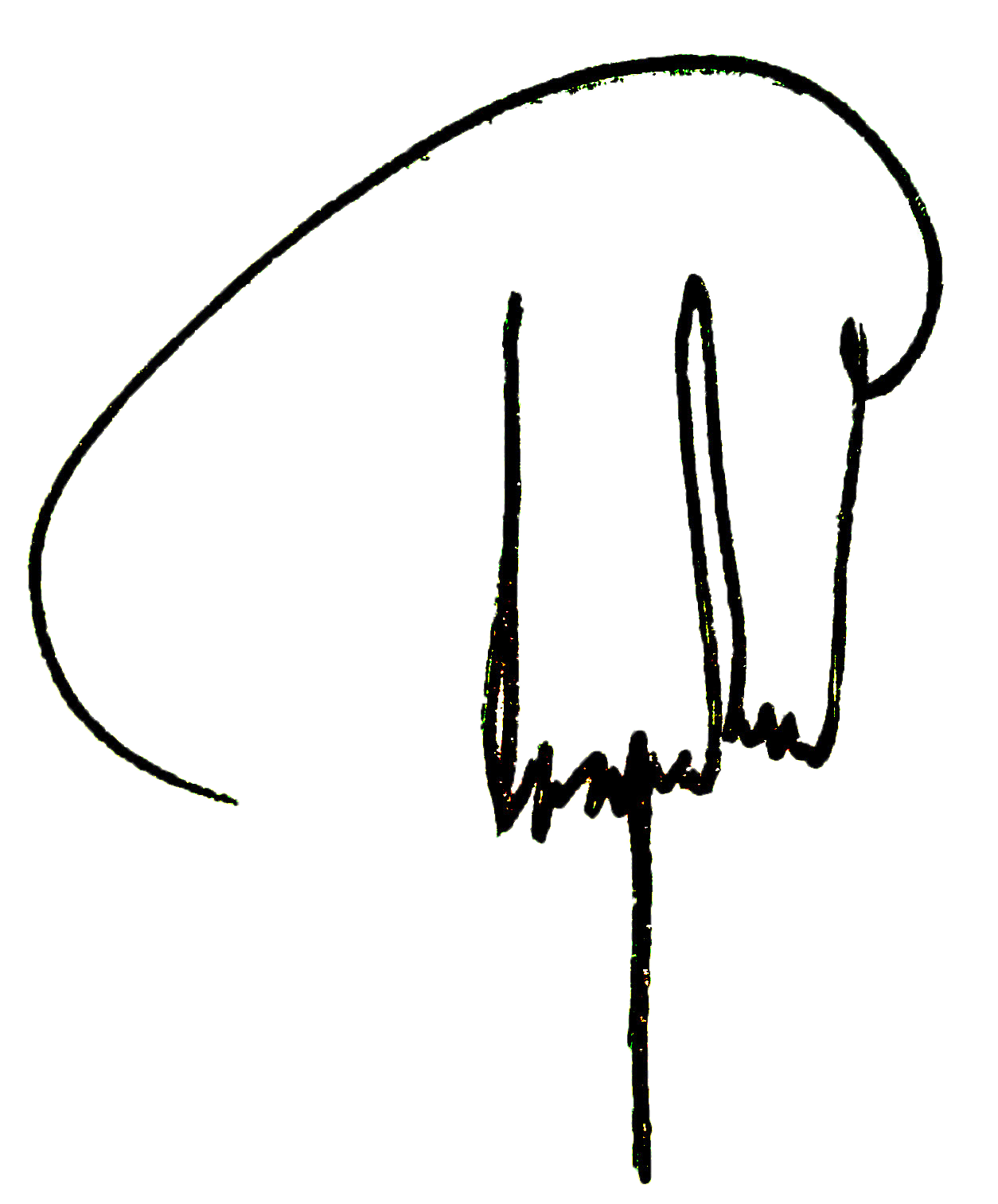
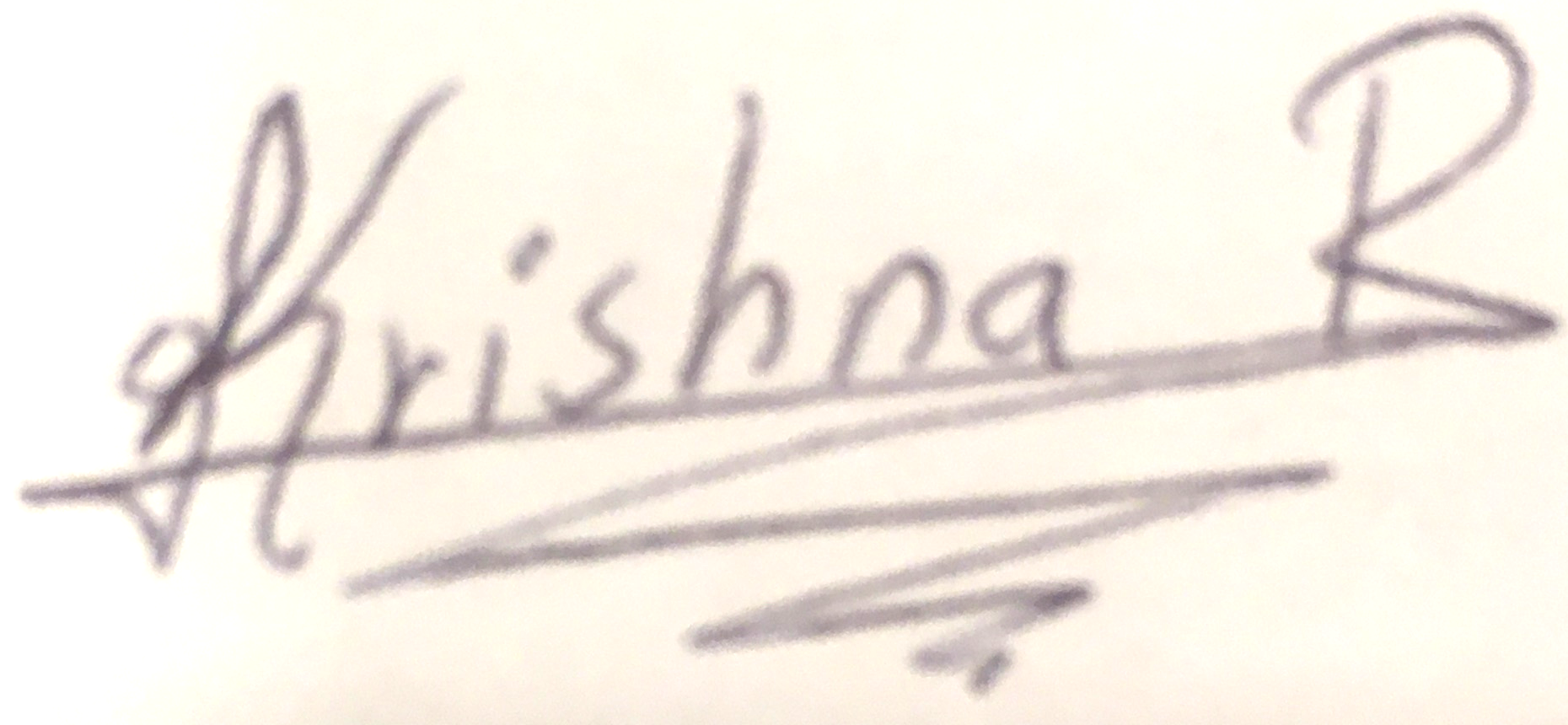
***System Architecture***

Object detection system would be developed on top of TensorFlow Object Detection API, and using SSD (Single Shot MultiBox Detector) Mobilenet:

SSD uses VGG16 to extract feature maps:



***AI Development Team*** 

Arianna Armijos Mateo Sebastian Lomas

Washington Pijal Krishna Roman