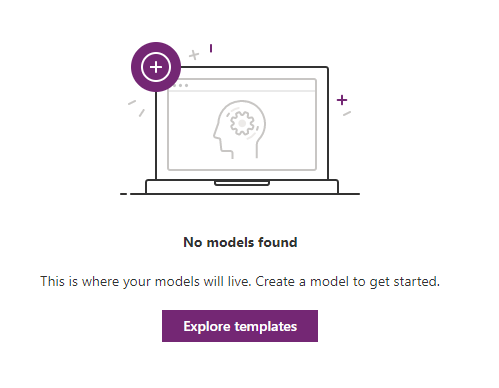
# Object detection

Object detection lets you count, locate, and identify selected objects within any image. You can use this model in PowerApps to extract information from pictures you take with the camera, or load into an app.

In this lab, we will build and train a detection model and build an app that uses the detection model to identify objects from available images.

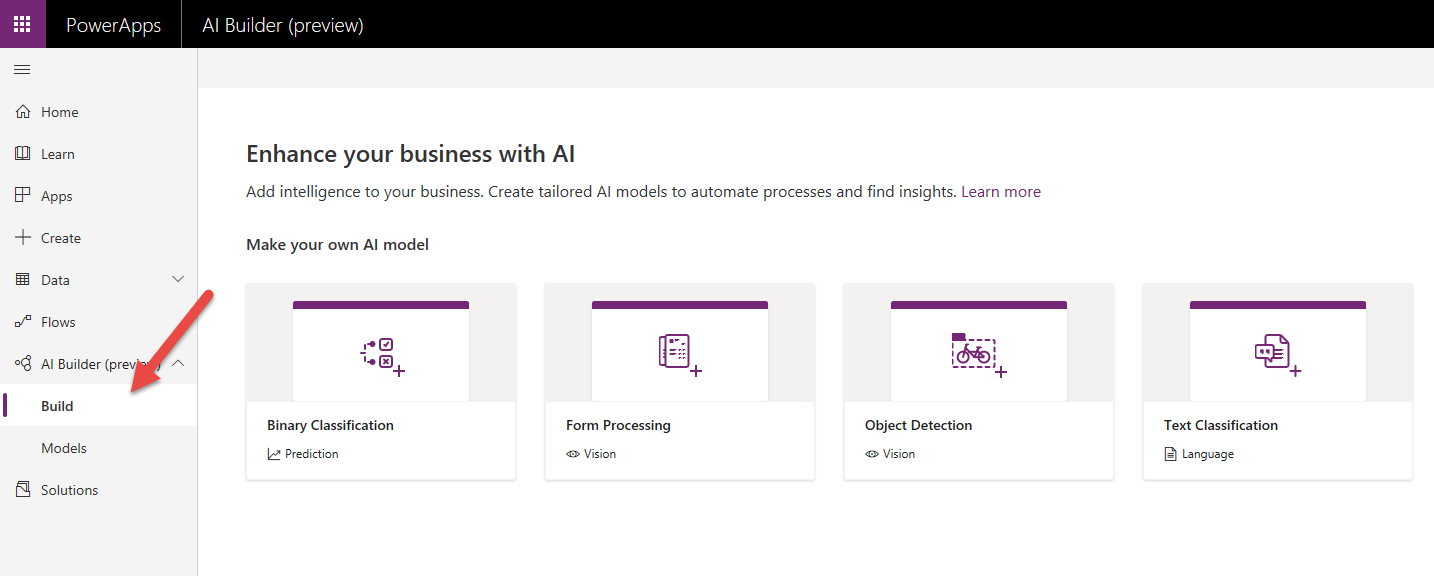
**Note:** If you are building the first model in an environment, click on Explore Templates to get started.



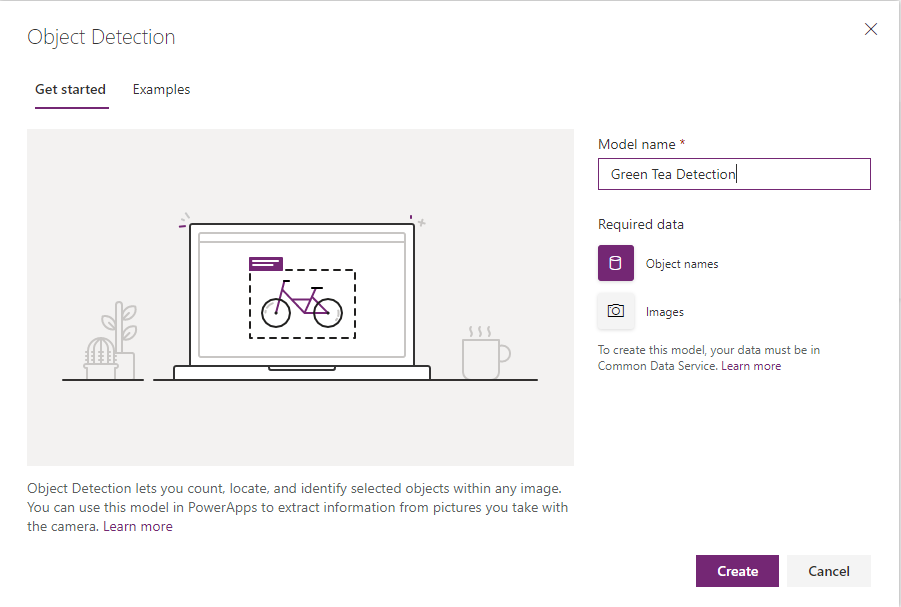
# Exercise 1

In this exercise we will build and train the Object Detection model for three varieties of tea.

1. In PowerApps maker, expand AI Builder and select Build. Select Object Detection.



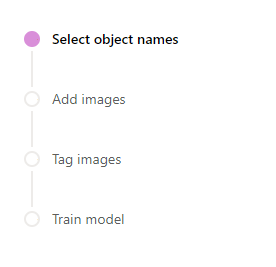
1. Name your model Green Tea Product Detection **and** because you are working in a shared environment also make sure to include your name as part of the model name. This will make it easier to find later. Click create.



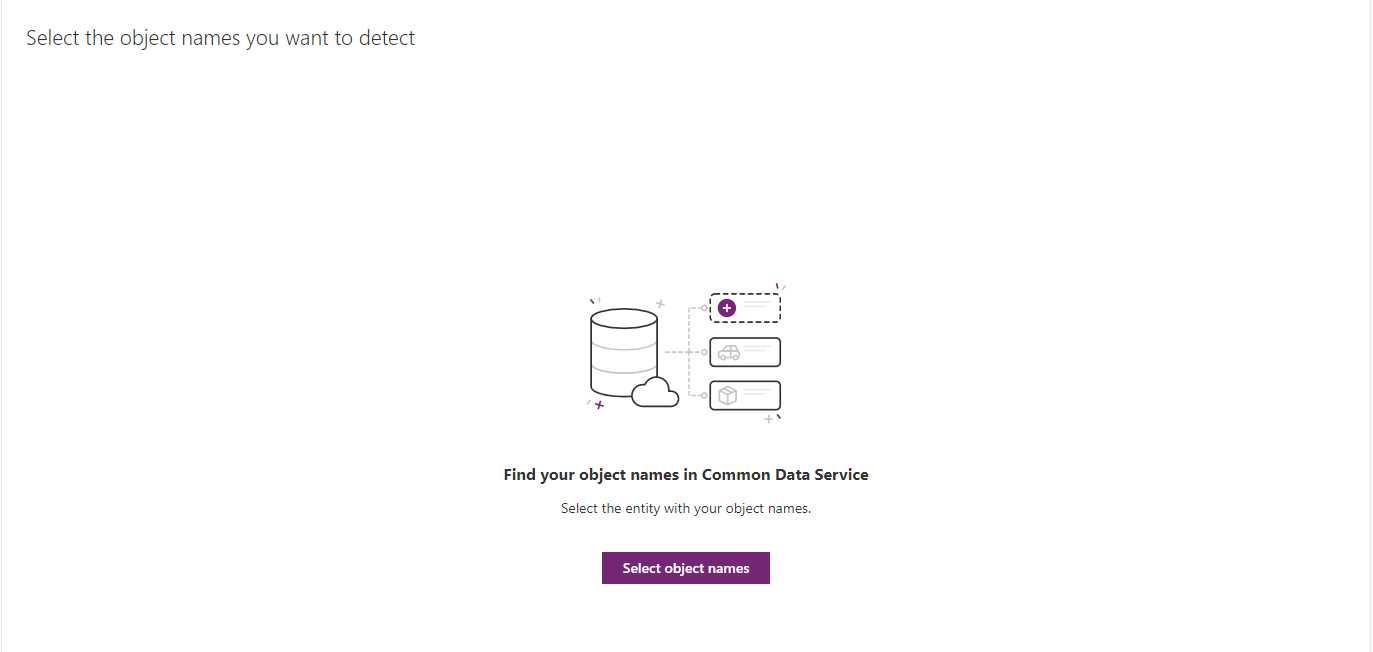
1. Your screen should now look like the image here.



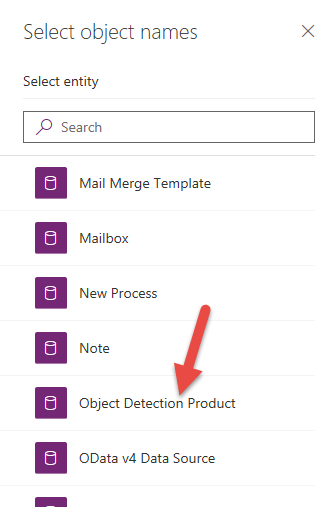
1. Notice the progress indicator on the left. Those are the steps we will follow now to build and train our model.



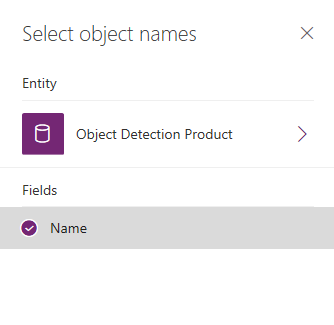
1. We are now going to define the objects we are tracking. Click on the Select object names.



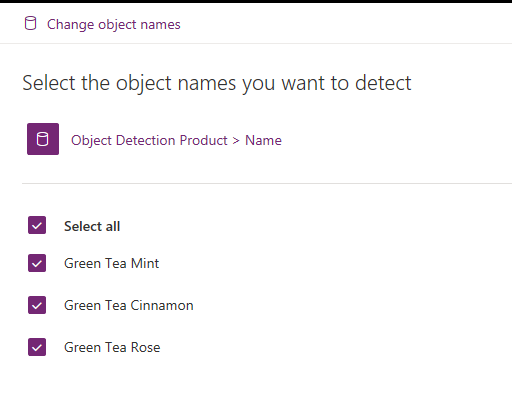
1. From the entity list, select Object Detection Product.



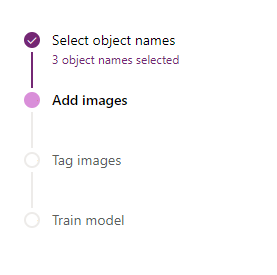
1. Select the Name field and click Select field.



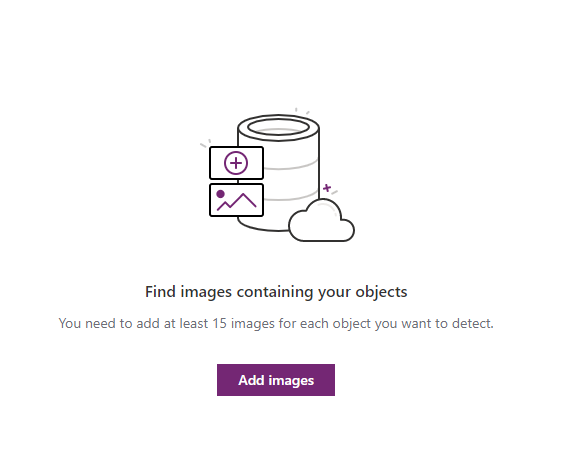
1. Select the tea items and click Next.



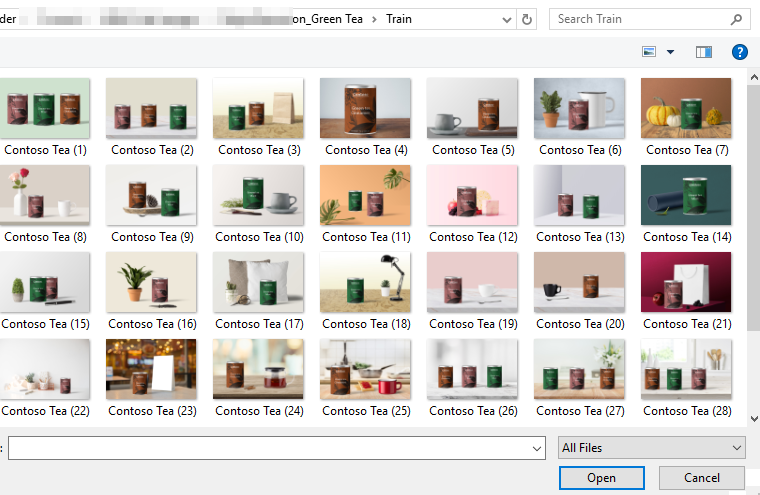
1. Notice the progress indicator has moved forward to the Add images step.



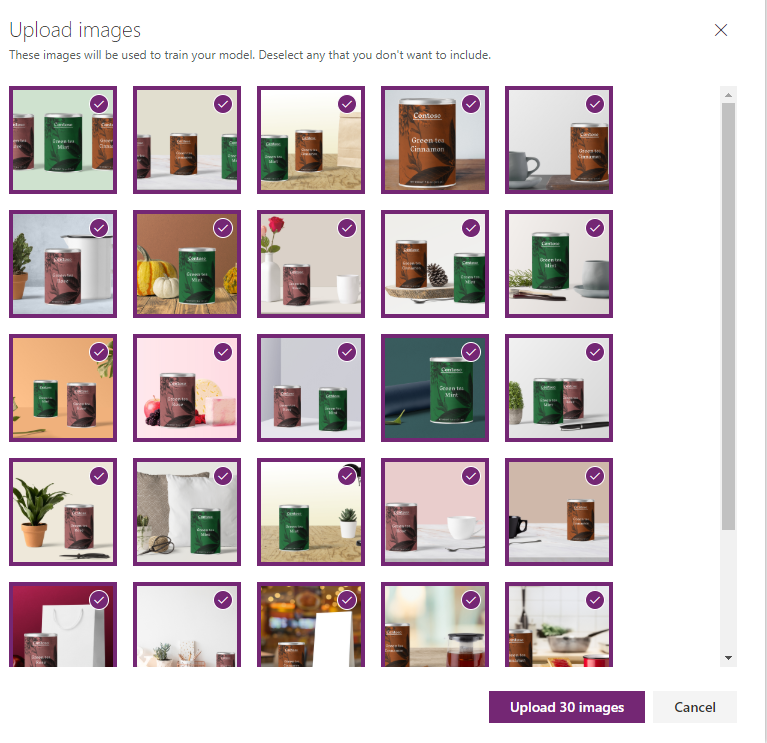
1. Click add images.



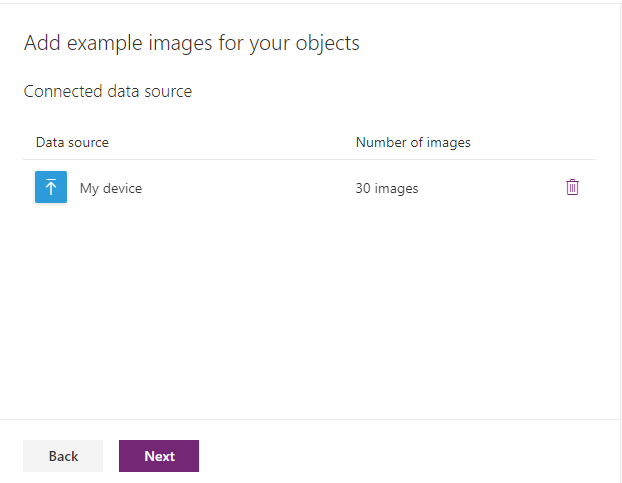
1. Select images from the set provided. You will need enough images to provide 15 samples for each type of tea we are tracking.



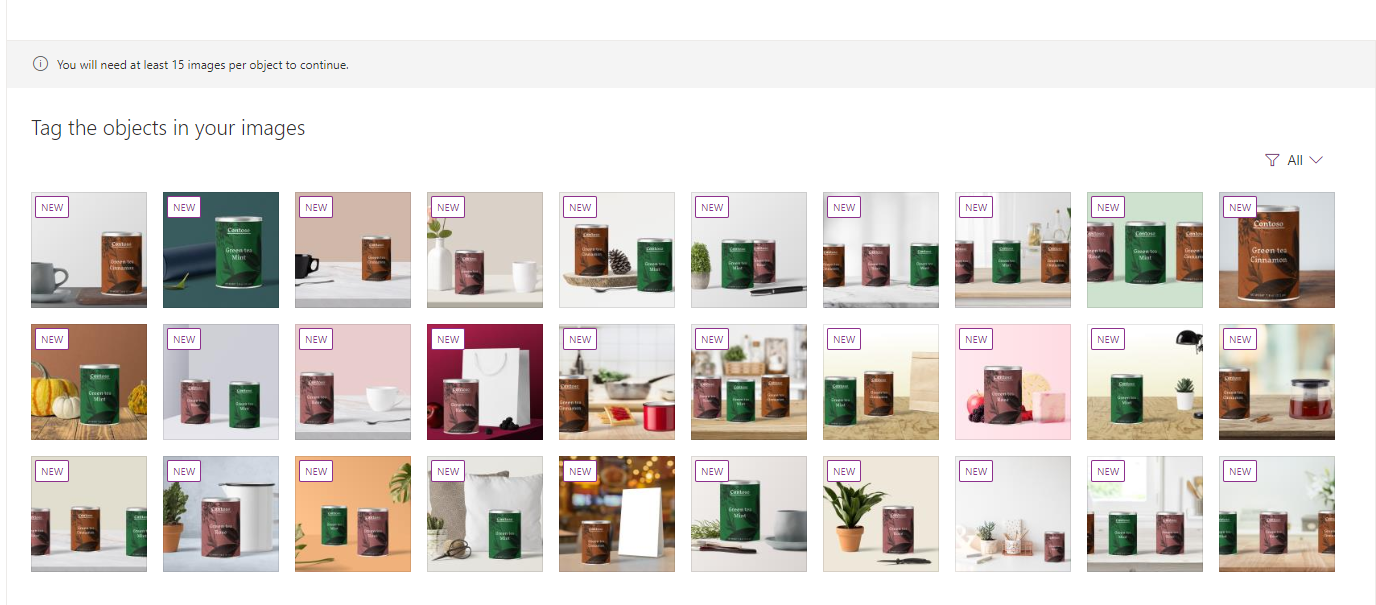
1. Approve the upload of images. Click Upload images. After the upload completes, click Close.



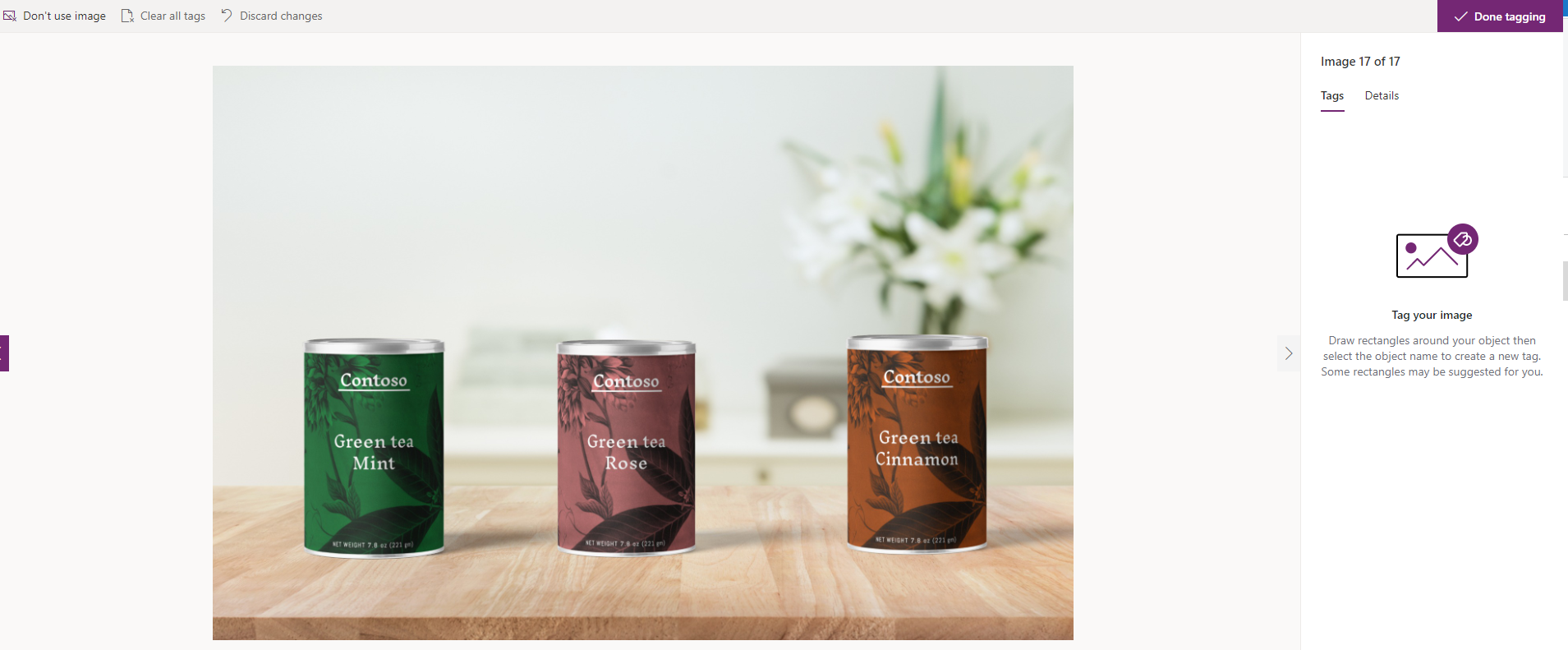
1. Click next to begin tagging the images.



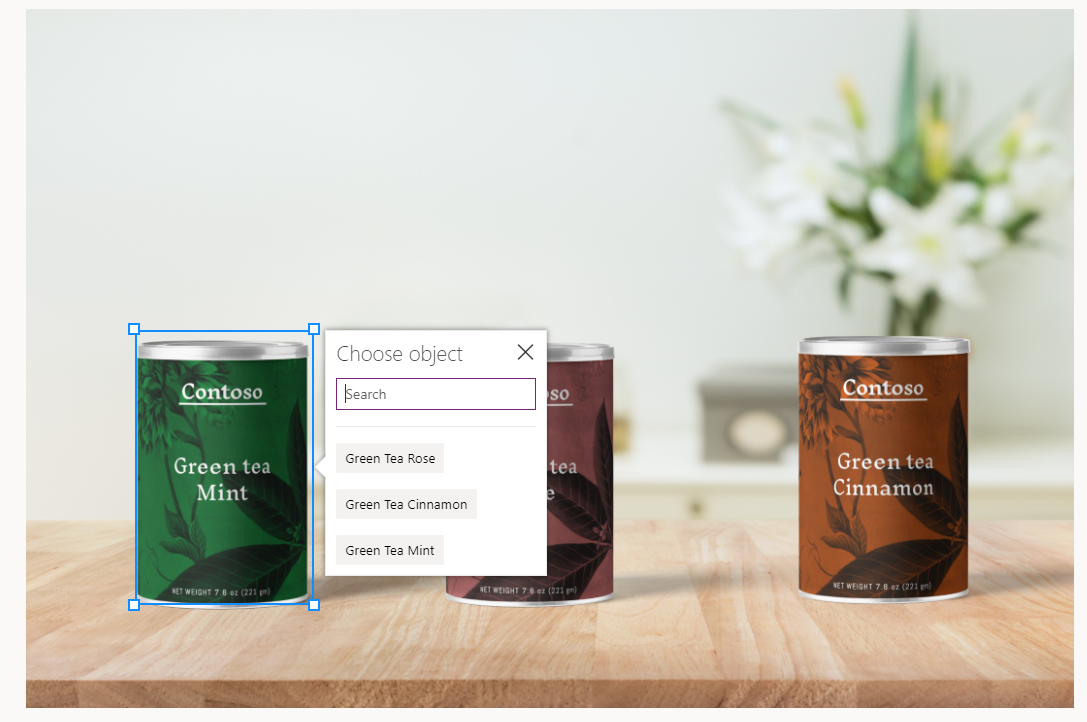
1. Select the first image to begin tagging.



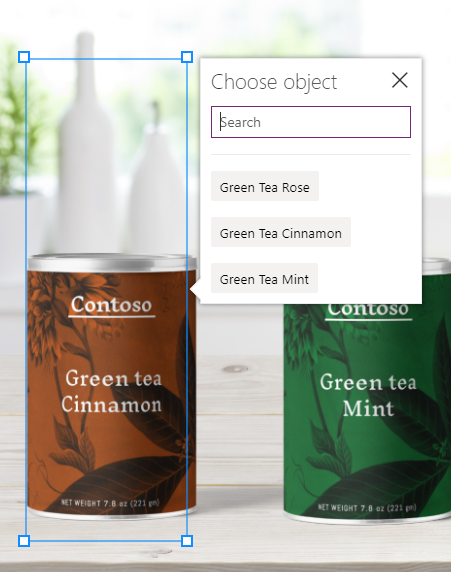
1. Hover over the image, near an item you wish to tag. A dotted-lined box should appear around the item. It has been detected as a single item that can be tagged.



1. Click on the item and select the matching object name.



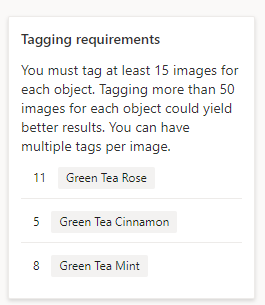
1. If the pre-defined selector is not accurate, as in the below example, you can drag the container to draw it to accurately tag the item.



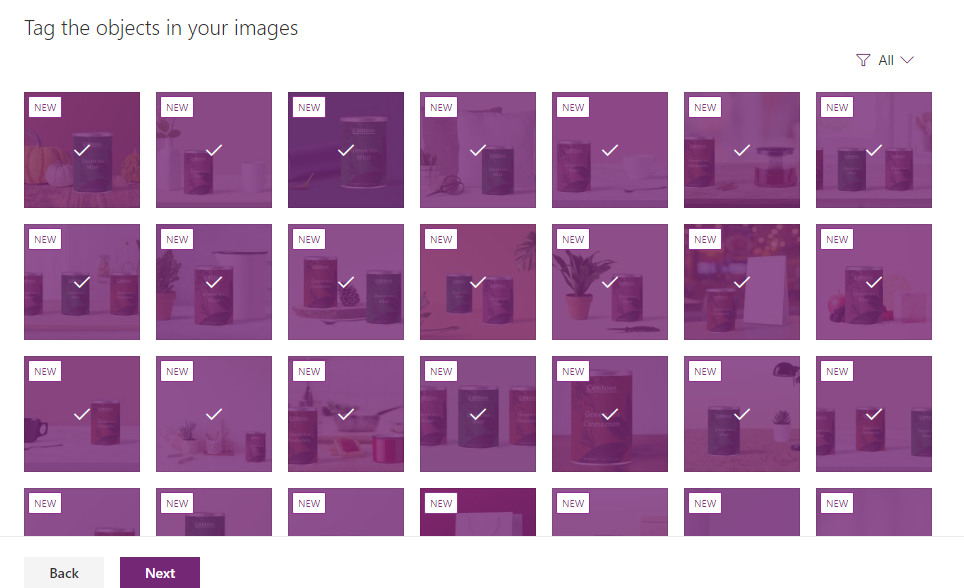
1. Do this for each item in the image and for each image in your set. When you have tagged all of the images you uploaded click Done Tagging in the top right of the screen.



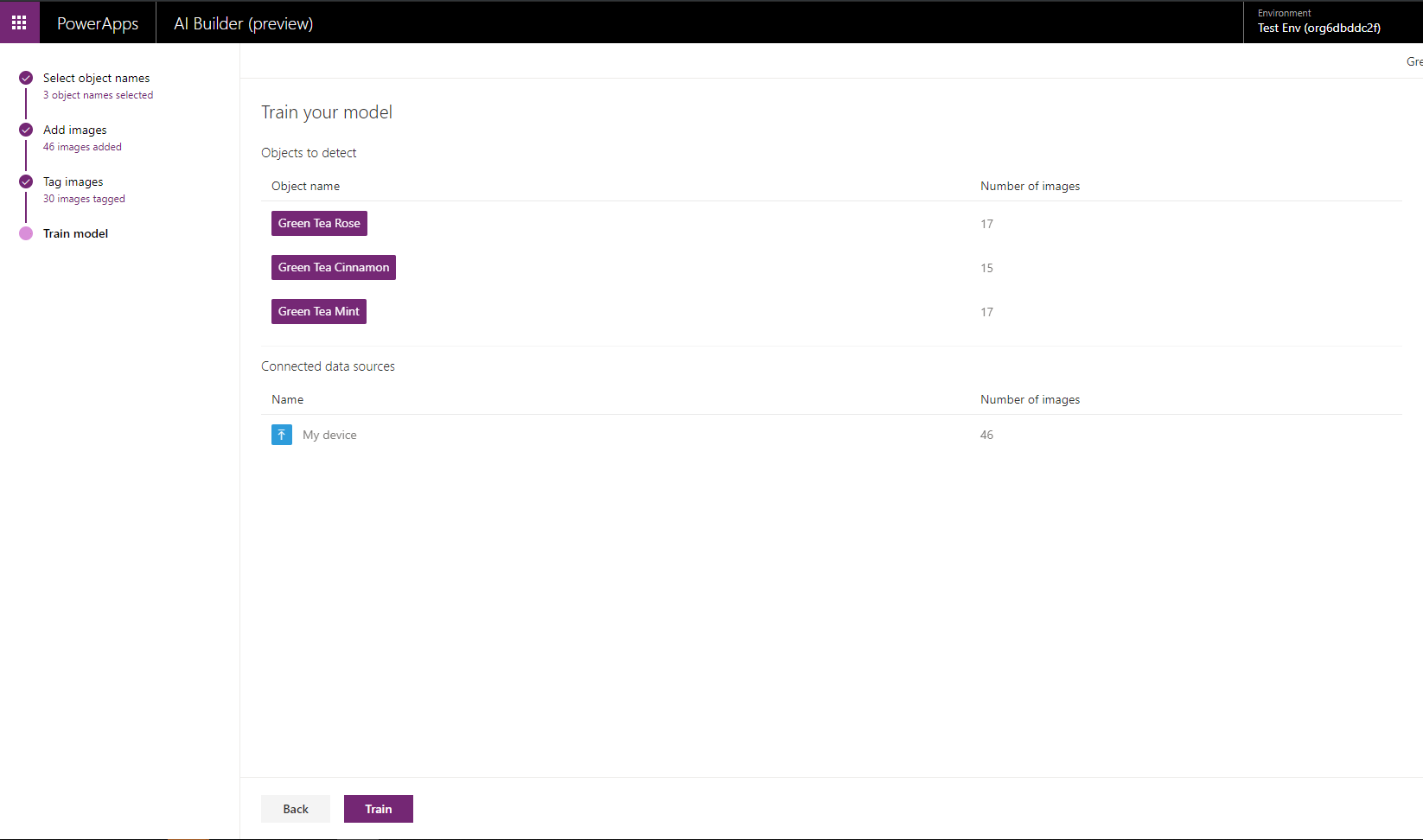
1. Once you have completed tagging, you will get a summary of the tags. If you haven’t tagged enough for analysis, you will need to load and tag more examples.



1. Once you have defined enough tags for training the model, you will be allowed to initiate the training. Click Next.



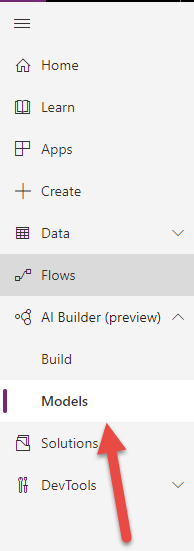
1. Click Train.



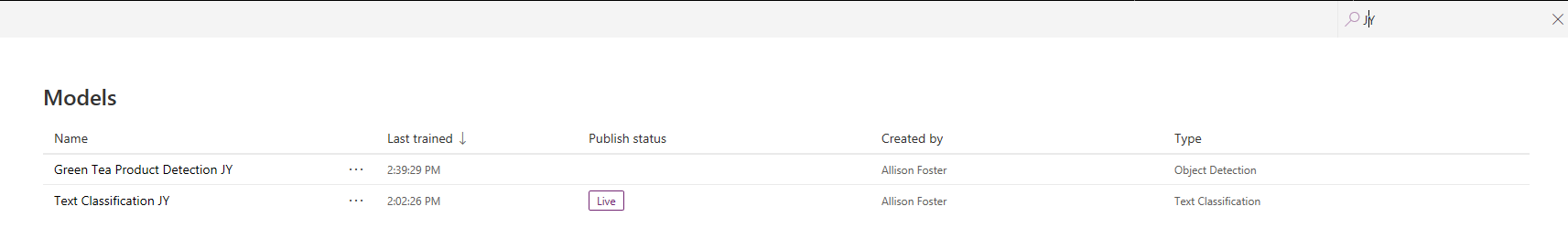
1. The training takes a few moments.



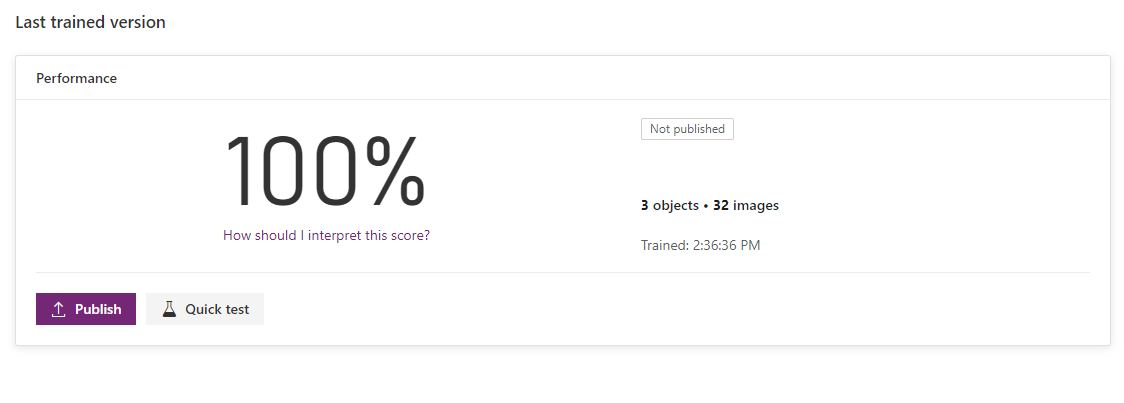
1. Navigate to the saved model view and confirm your model has completed training.

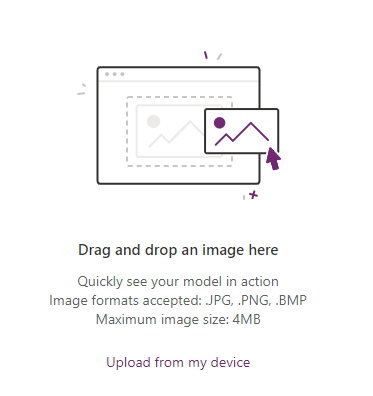


1. Select the model you just made.



1. Select Quick test.



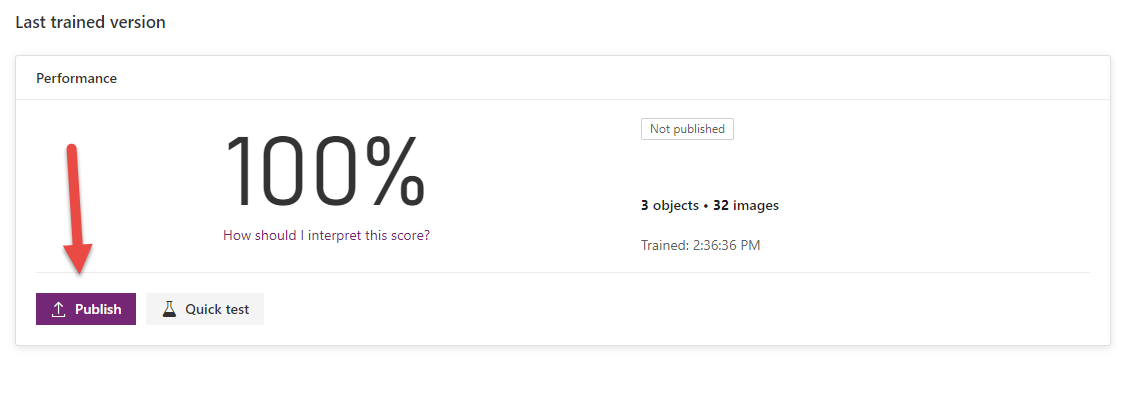
1. Upload or drag and drop one of your test images to be analyzed.   
     
   
2. You will see the analysis and level of confidence for the match.



1. Upload an image you know will not match. You will see the analysis and level of confidence for the match.



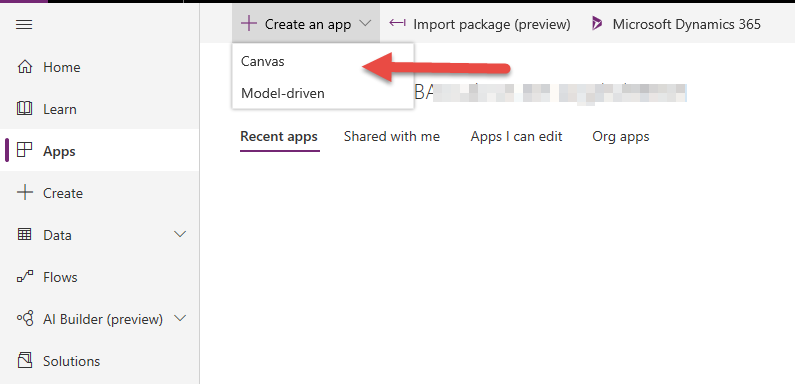
1. Click close.
2. Publish your model.



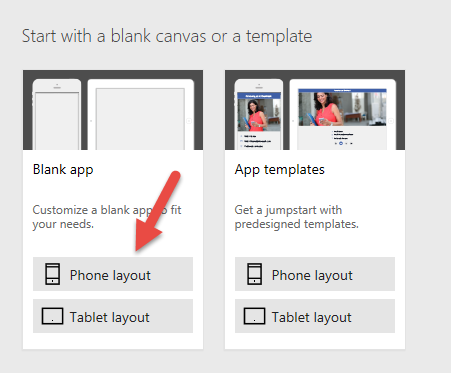
# Exercise 2

We will now create a canvas app you can use for detecting the items that have been trained in our model. The product will be detected from the image and you will be able to adjust on-hand inventory for the item.

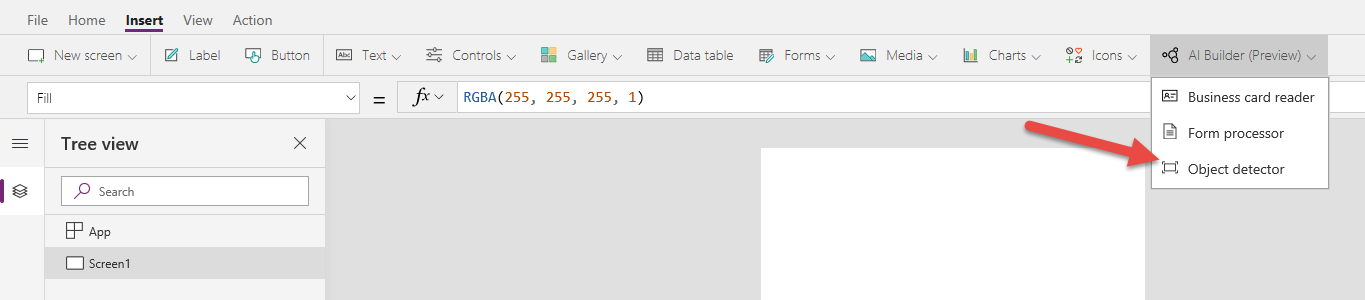
1. Navigate to Apps, and select Create an app, then select Canvas. If asked, grant permission for the app to use your active CDS credentials.



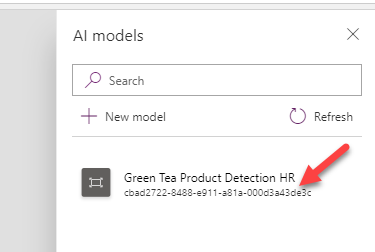
1. Select Blank app with Phone layout.



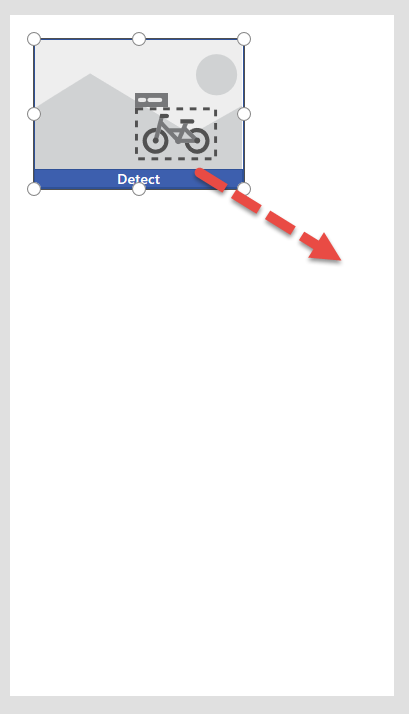
1. On the maker canvas, select the Insert tab in the ribbon and expand AI Builder. Select Object detector to place this control on your app.



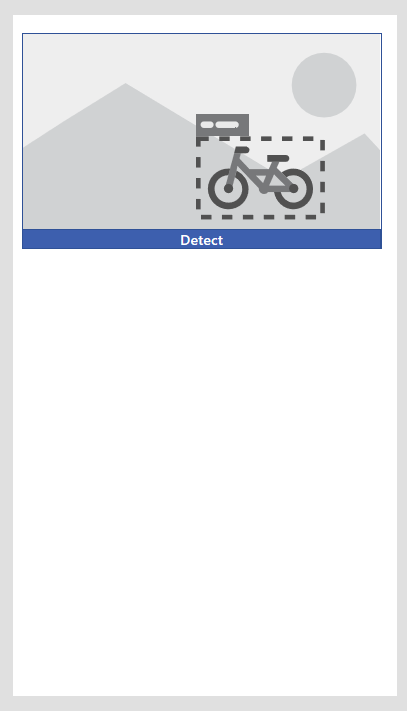
1. Select the AI model you built.



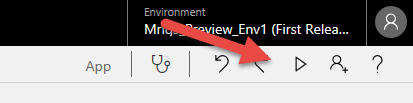
1. Resize the control to better use the space.



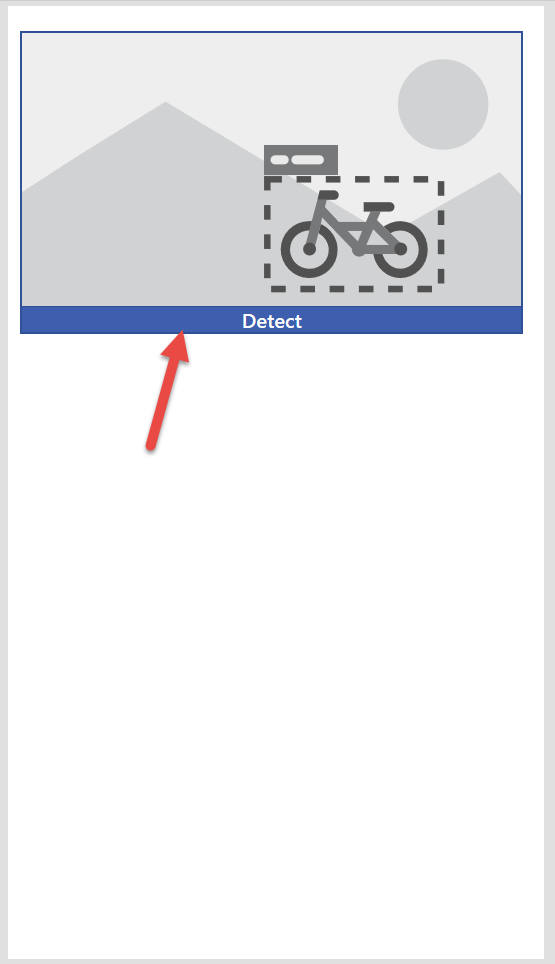
1. Make sure to leave room for more items we will be placing soon.



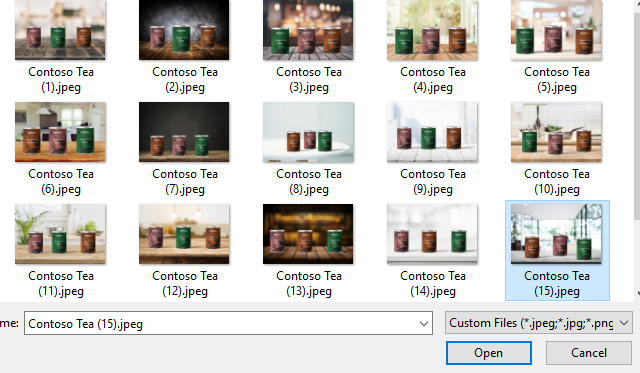
1. Play your app.



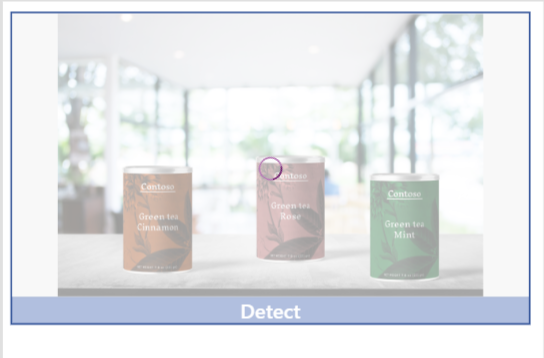
1. Click on Detect.



1. Choose one of your test images and click Open.



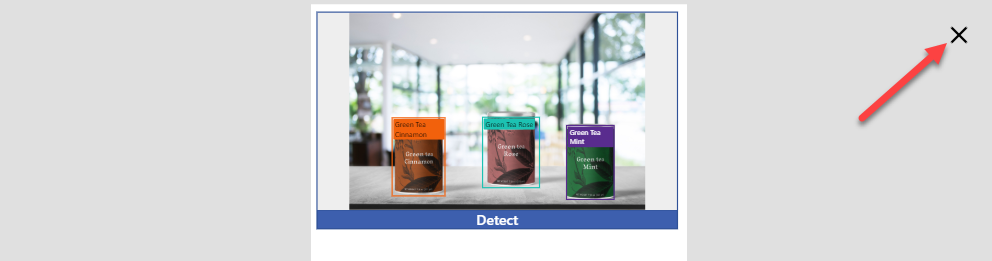
1. The image will now be analyzed.



1. Our model has detected each tea in the image.

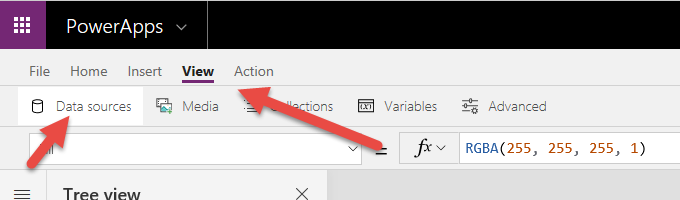


1. Exit the app player.

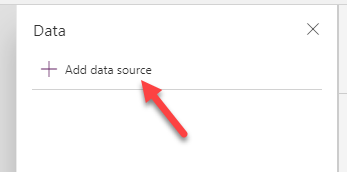


## Bonus exercise- build out the data in your canvas app

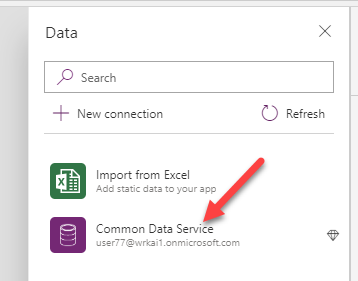
1. We will now select our data source. Select View from the ribbon and select Data Sources.



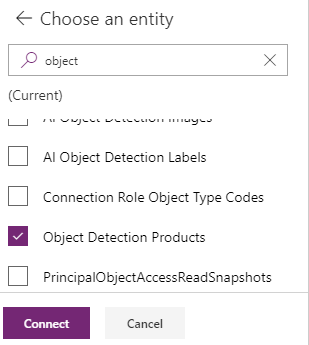
1. Click + Add Data Source.



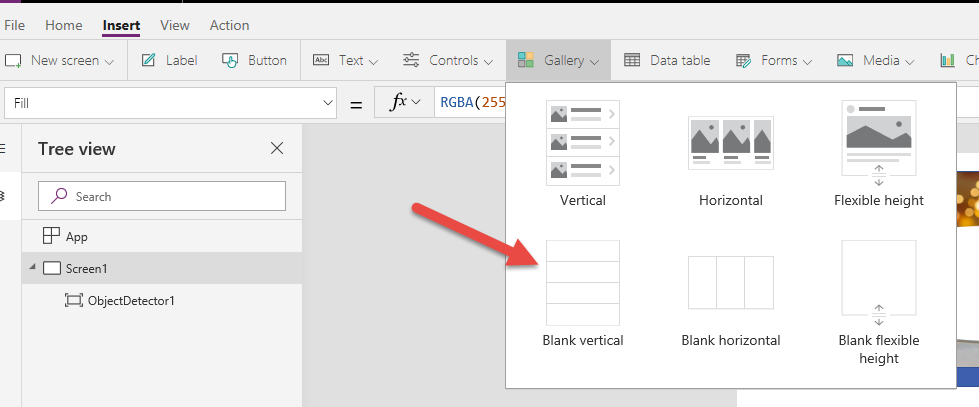
1. Add the Common Data Service data source. **Do not use** Common Data Service (current environment).



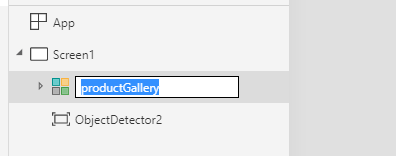
1. Select the Object Detection Products entity and click Connect.



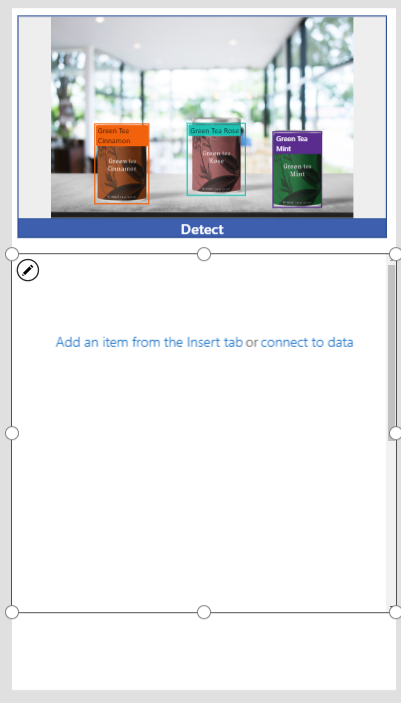
1. Close the Data pane.
2. With Screen1 selected in the Tree view, navigate to the Insert ribbon tab, expand Gallery and select Blank vertical gallery.



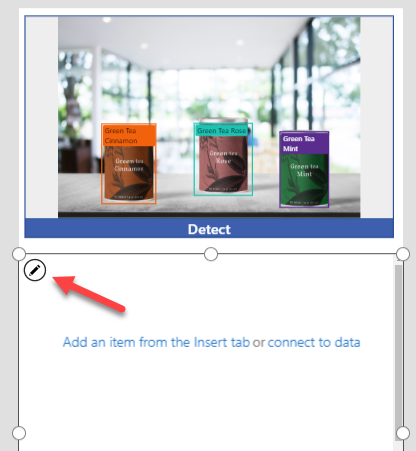
1. Rename the Gallery productGallery. You are re-naming the gallery so you can reference it from your formulas.



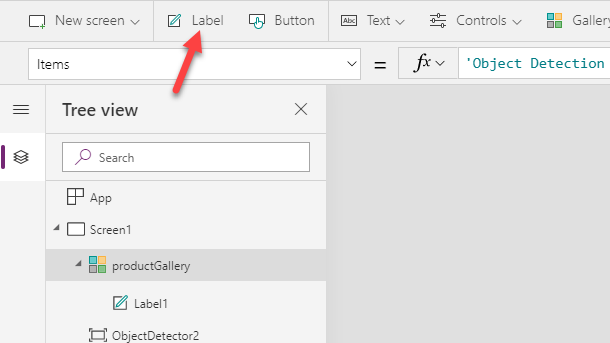
1. Resize and move the gallery control to fit the available space on the screen, leaving some space at the bottom for using later.



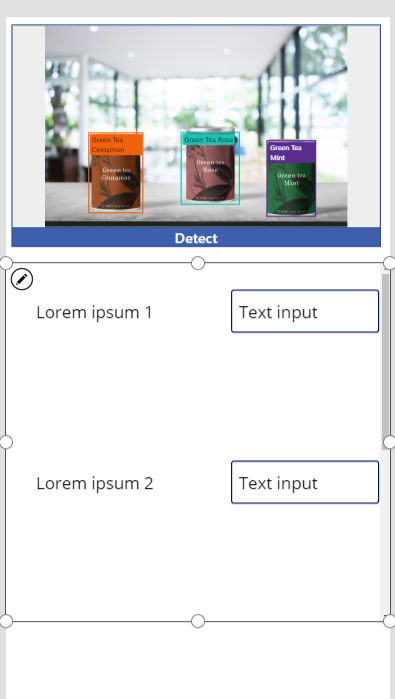
1. Select the edit icon from the gallery.



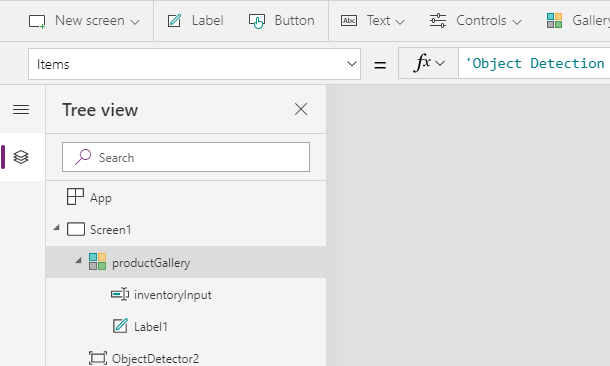
1. Add a label to the gallery.



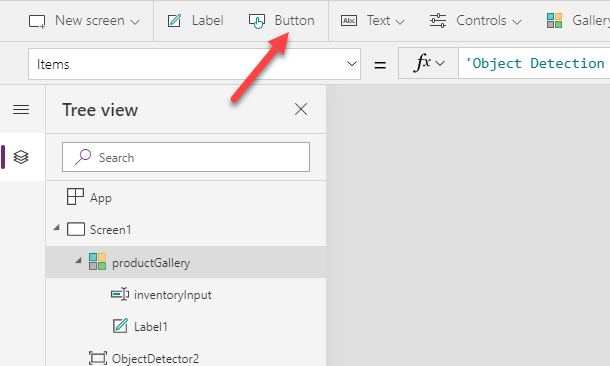
1. Click edit again and add a Text input box to the gallery. Resize and place it to line up with the label we’ve already placed. We will be updating inventory counts in this text box.



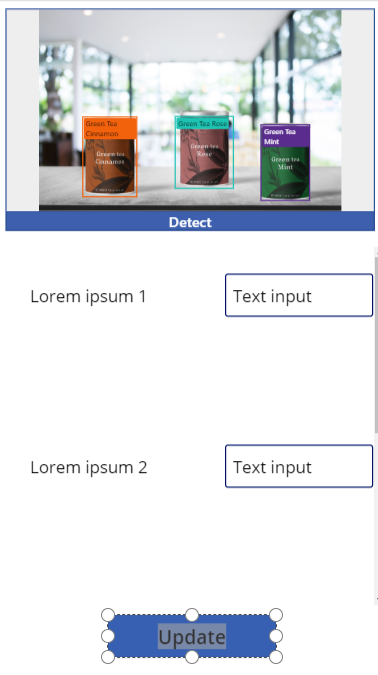
1. Rename the Text Input inventoryInput. You are renaming this control so you can reference it from your formulas.



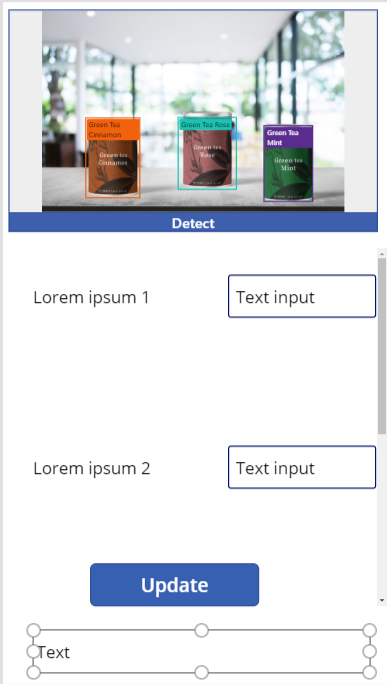
1. With focus on the Screen1 in the Tree view, click in the ribbon on Insert and select Button.



1. Drag and move the button to the bottom of the screen, double click on it to edit the text. Rename it to Update.

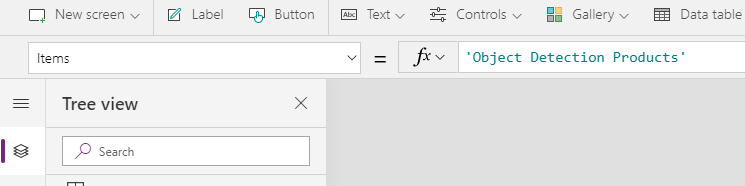


1. We will now add the user message to give the user confirmation their submission was accepted; we will define this logic later. With focus on Screen1, insert a label, drag it to the bottom of the screen.



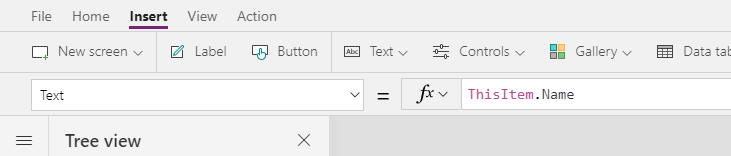
1. We will now add logic to the controls we’ve placed on the screen. Select the Gallery and replace the Items formula with the following.

'Object Detection Products'



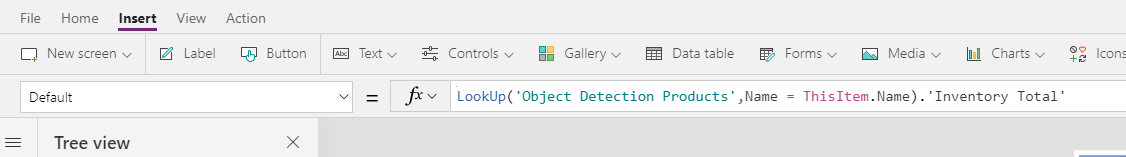
1. Select the label in your gallery. Replace the Text formula with the following:

ThisItem.Name



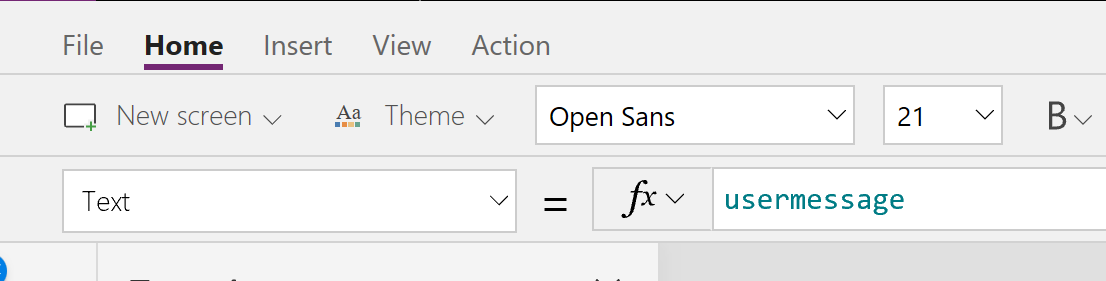
1. Select inventoryInput and replace the formula for Default with the following:

LookUp('Object Detection Products',Name = ThisItem.Name).'Inventory Total'

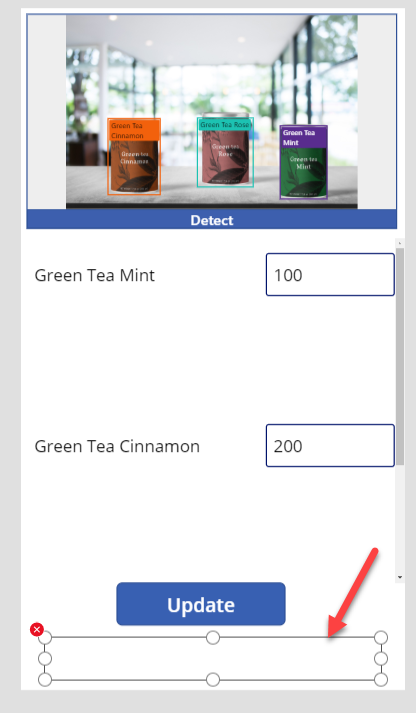


1. Select the other label (the one that shows at the bottom of the screen) and replace its text with the following:

usermessage

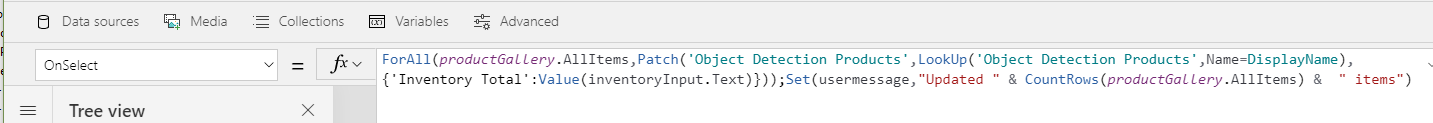


1. You’ll notice that area now looks blank. We will configure that message in our next step.

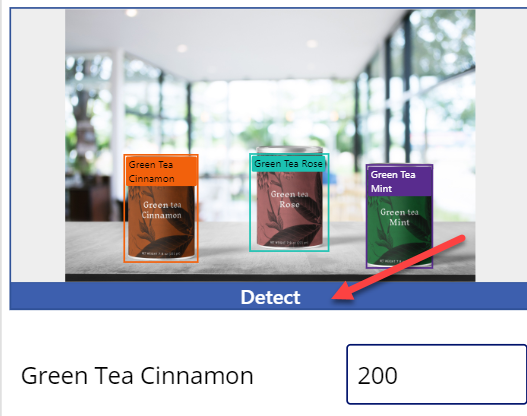


1. Select the button control and replace the OnSelect with the following:

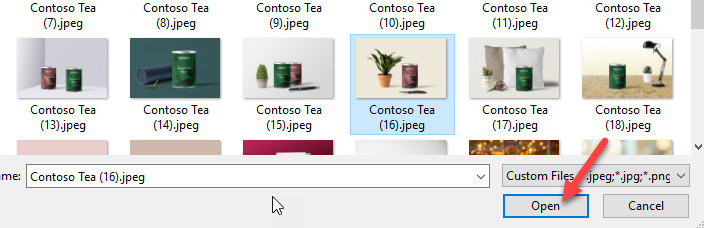
ForAll(productGallery.AllItems,Patch('Object Detection Products',LookUp('Object Detection Products',Name=DisplayName),{'Inventory Total':Value(inventoryInput.Text)}));Set(usermessage,"Updated " & CountRows(productGallery.AllItems) & " items")



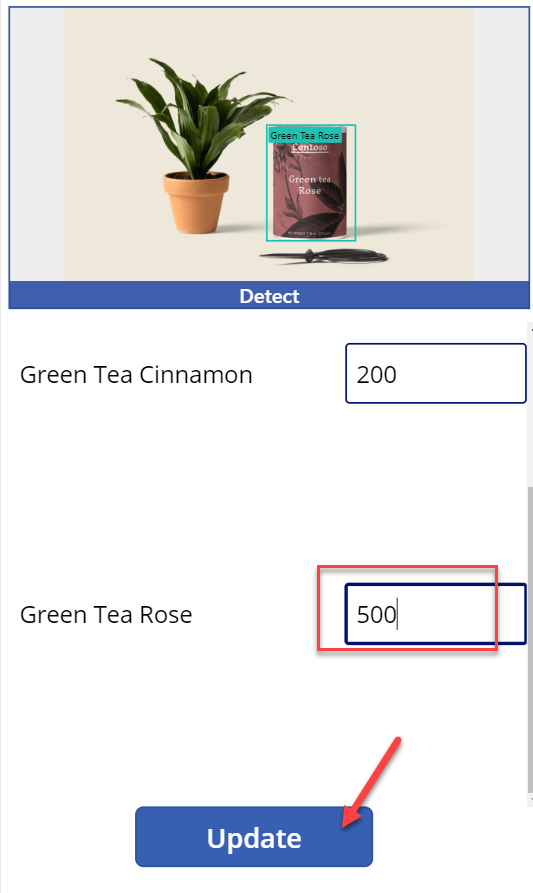
1. Play the app again.
2. Click Detect.



1. Select an image to evaluate.



1. Update the quantity for the correct product and click Update.



1. The bottom should show a message now.

