

The image displays the MIT App Inventor web interface, showing the design and logic views of an application named "PhotoReconnection".

Design View (Top):

- Left Panel (Component Palette):** Lists various UI components under "Interfaz de usuario", including Botón, CasillaDeVerificación, CircularProgress, SelectorDeFecha, Imagen, Etiqueta, LinearProgress, SelectorDeLista, VisorDeLista, Notificador, CampoDeContraseña, Deslizador, and Desplegable.
- Center Canvas:** Shows a mobile app preview with a blue header "Screen1", a globe icon, and a red "Clasificar" button at the bottom.
- Right Panel (Design View):** Shows the "TeachableMachineImageClassifier" component, with options to "Cambiar nombre" or "Borrar".

Logic View (Bottom):

- Left Panel (Component Palette):** Lists various logic components under "Integrados", including Control, Lógica, Matemáticos, Texto, Listas, Diccionarios, Colores, Variables, and Procedimientos.
- Center Canvas:** Shows the logic blocks for the "Clasificar" button click event. The logic is as follows:
 - When "ClasificarBoton1" is clicked, call "TeachableMachineImageClassifier1" to "ClassifyVideoData".
 - When "TeachableMachineImageClassifier1" returns "GoClassification", execute the following:
 - Set "VBrenda" text to "Brenda" (rounded).
 - Set "VSoleo" text to "Soleo" (rounded).
 - Obtain the value for the key "Brenda" in the dictionary. If not found, set "Resultado" to "not found".
 - Obtain the value for the key "Soleo" in the dictionary. If not found, set "Resultado" to "not found".
- Right Panel (Logic View):** Shows the "TeachableMachineImageClassifier" component, with options to "Cambiar nombre" or "Borrar".

The interface includes a top navigation bar with "PhotoReconnection" and "Donar a App Inventor" buttons, and a bottom status bar showing the system clock and temperature.

How does it work?

Machine learning models are trained on examples (e.g., images, sounds, poses) gathered by the creator. Their results depend on the data they've been trained on.

Want to use this model in your project?

See [this link](#) to learn how to use Teachable Machine models in your projects.

Report this model:

If you have concerns about this model, report it using [this form](#).

This model:

teachablemachine.withgoogle.com/models/WNRXYW7vMc/

/model.json

The model architecture, used by TensorFlow.js library

/metadata.json

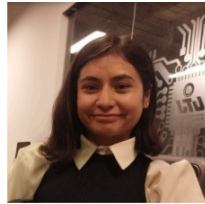
Contains the model metadata, for example class labels and version of library

/model.weights.bin

TensorFlow.js binary file containing the model weights

or drag & drop here

Import images from Google Drive



↓

Output

Brenda

100%

Sóleo

HOW DOES IT WORK?

Machine learning models are trained on examples (e.g., images, sounds, poses) gathered by the creator. Their results depend on the data they've been trained on.

Want to use this model in your project?

See [this link](#) to learn how to use Teachable Machine models in your projects.

Report this model:

If you have concerns about this model, report it using [this form](#).

This model:

teachablemachine.withgoogle.com/models/WNRXYW7vMc/

/model.json

The model architecture, used by TensorFlow.js library

/metadata.json


Contains the model metadata, for example class labels and version of library

/model.weights.bin

TensorFlow.js binary file containing the model weights

or drag & drop here

Import images from Google Drive



↓

Output

Brenda

13%

Sóleo

87%