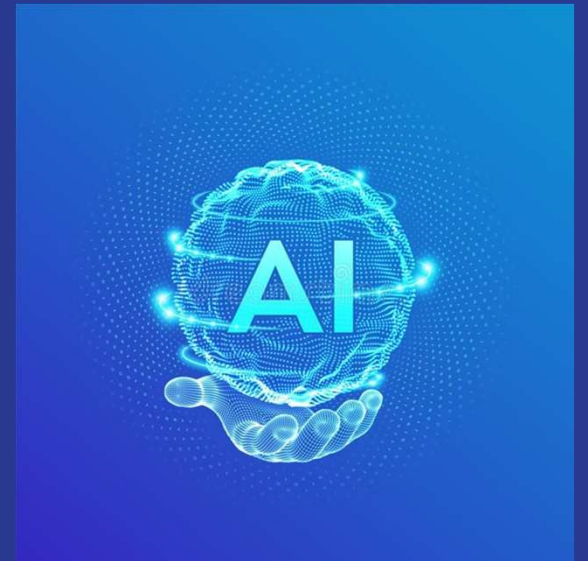


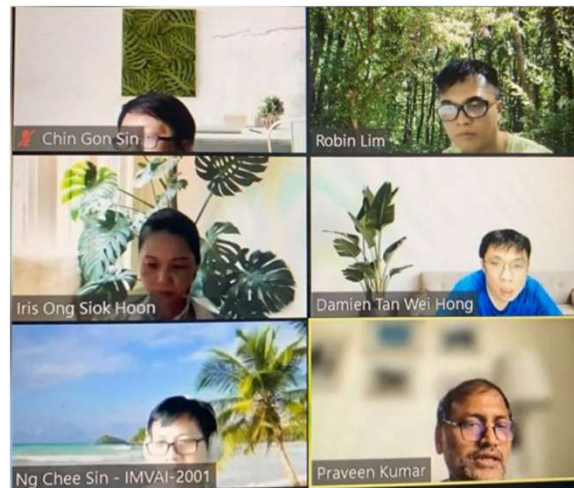
# AI Practitioner Final Project Presentation

Team 3



# ***AI Practitioner Project Team Members***

- Praveen Kumar
- Ong Siok Hoon (Iris)
- Chin Gon Sin
- Ng Chee Sin
- Robin Lim
- Tan Wei Hong

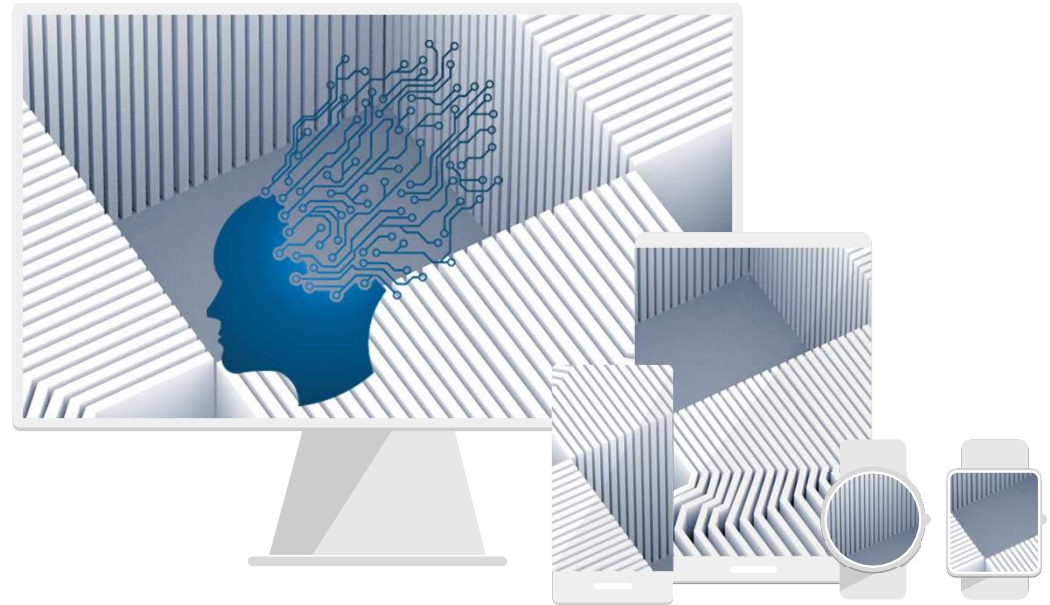


# ***TEAM3***



# Agenda

- Team Members
- Project Topic
- Enterprise Design Thinking (EDT)
- AI Essentials Framework
  - Intent
  - Data and Policy
  - Understanding
  - Reasoning
  - Knowledge
- Project Planning & Design
  - Integration
  - Dog Identification AI
  - Chatbot
- Project Demo
- Project Learnings/Challenges



# Project Topic

## Dog Lovers Paradise

### IDENTIFY

*Shows us a photo of the dog that you like! And we will do the rest!*



[Click to open Dog Breed Classifier tool](#)

#### Identify Dog Breed

*Upload any dog photo and get an instant dog breed identification using our AI tool.*



[Click to open wikiHow Pet Dog Basics](#)

#### Taking Care of dogs

*New to dog ownership and need to learn the basics? Find answers to dog care questions.*



[Click to open video on Healthy Feeding](#)

#### Dog's diet

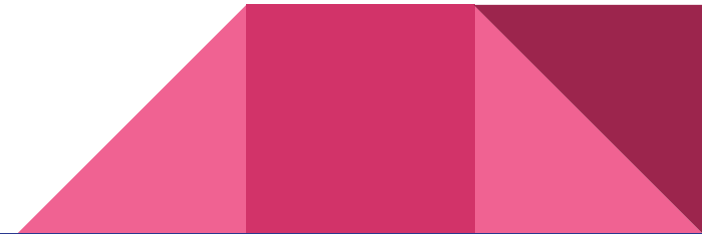
*Know more about what food should a healthy dog takes.*

\* Disclaimer: For the purpose of this final project for IMVAI-2001, our dataset does not cover all the breeds as the total size of all the breeds is outside the free storage capacity of online app deployment platform.

**Getting bored?** Come have a dog chat! **Woof Woof!** 🐕

## Introduction

# Enterprise Design Thinking (EDT)



# Persona

**Our Passionate Dog Lover - Charlie**



**Age - 35 , Single**

**Based in SG , graduated with a  
College Degree in IT**

**Now Working as an IT Engineer**

# Persona

## Motivations

---

Enjoy  
meeting  
other dog  
lovers

Want to  
know more  
about dogs

Always  
interested in  
what technology  
can do for the  
community

## Goals

---

Meet like  
minded dog  
lovers

He hopes to  
bring in  
different  
breeds of dog  
to his existing

## Needs

---

He would love  
to meet friends  
with common  
interests like  
dogs.

Needs to get  
more  
information  
about dogs  
before his next  
adoption

He always panic  
and felt helpless  
when his dog was  
not right on many  
occasions.  
Desperate for quick  
tips or advices.



# Pain Points

Ever came across irresponsible pet owner who keeps a husky without knowing much of its habits and interests.

Find it difficult to identify the different breeds of dogs since there are so many of them

How do I feed my dog with healthy food and train them well ?

Stress and anxiety when I feel that the dog is not right today. Panting or excessive licking, what should I do ?

Thinking if there is an integrated platform for him to find out more about dogs and make new friends with the common interest

Can I cut their nails myself ?  
How do I go about doing that ?



# Enterprise Design Thinking

## Problem Statement

How Might We **Come out with an interactive project**

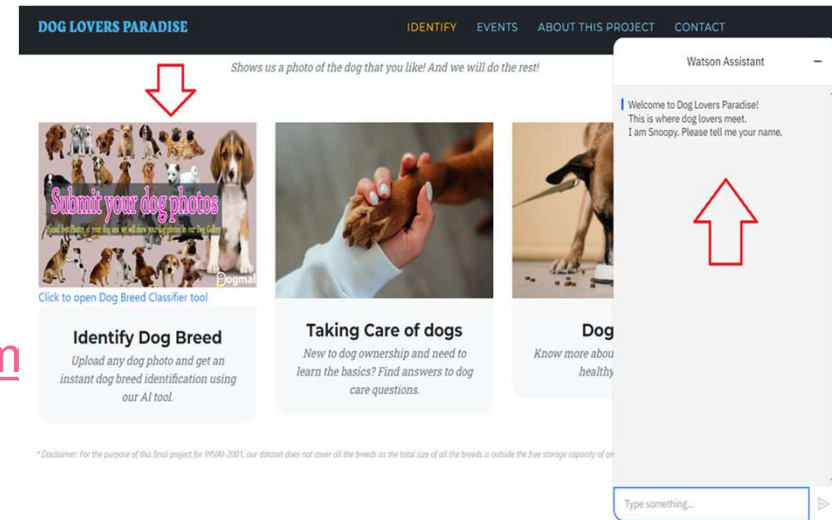
For **Dog Lovers**

So that **They can gain more knowledge about dogs & meet like minded people**



# A Special Gift For All Dog Lovers

[Dog Lovers Paradise \(t3dogbreeds.herokuapp.com\)](https://t3dogbreeds.herokuapp.com)



- Our Dog identification feature - Simply just upload any dog pic and we will tell u what is the breed !!
- Find out more about your dog breed with our Snoopy chatbot. Keep up with any latest events !!!

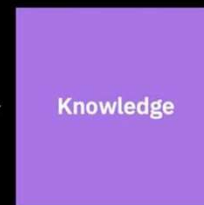
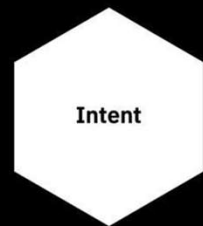
# AI Essentials Framework

## AI Essentials Framework

Business

World

Machine



Why are we doing this and what are the big ideas that support the effort?

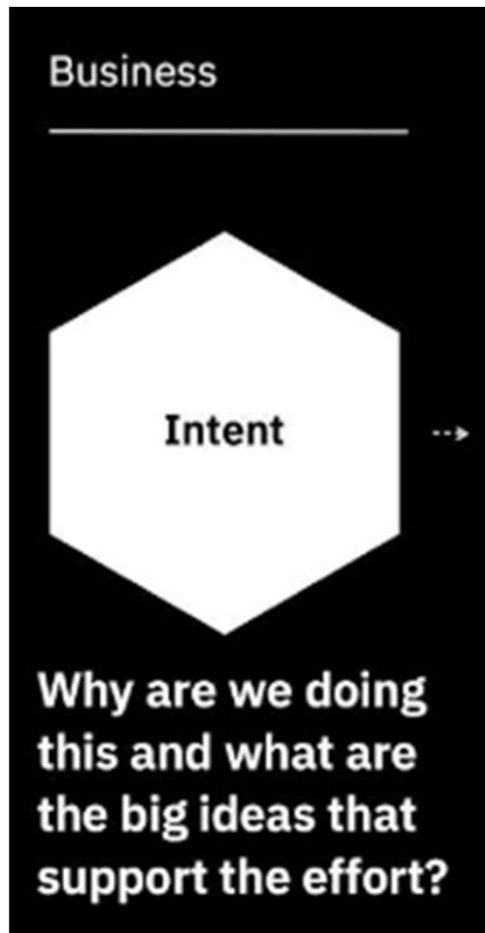
What data do you have to bring these ideas to life?

What will a machine need to understand from this data so it can reason?

How will a machine apply logic to what it knows?

If you succeed, what are the implications of what your AI will know?

# Business: Intent



# Intent

A community that gathers dog lovers from all walks of life.

1. Identify dog breed
2. Updates of upcoming doggie events
  - Doggie Walkathon
  - Doggie Grooming
  - Doggie Care
3. Chatbot for queries
4. Tinder for dogs and even for human (perhaps ><)

## 6 core AI intents

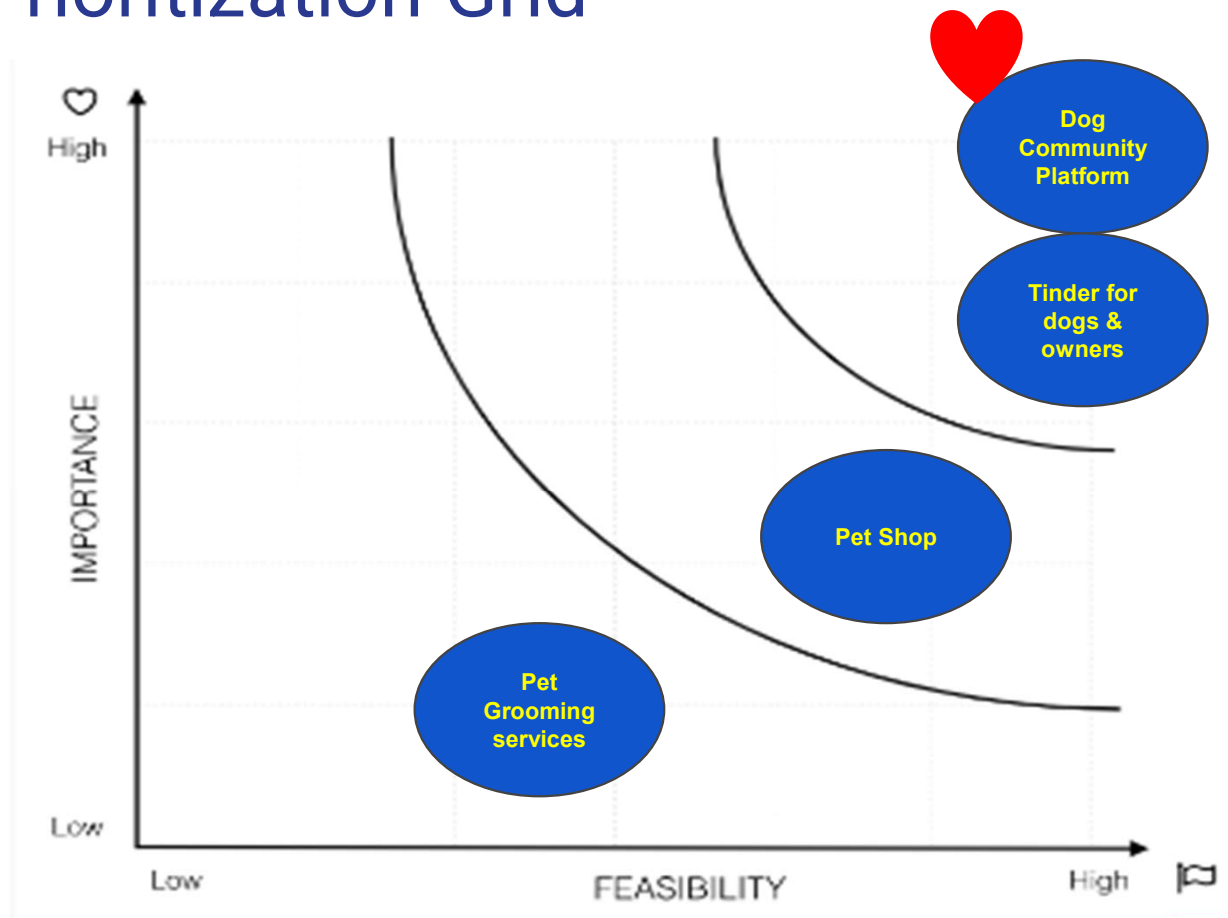
- Accelerate research and discovery
- Enrich your interactions
- Anticipate and preempt disruptions
- Recommend with confidence
- Scale expertise and learning
- Detect liabilities and mitigate risk

# Big Ideas



1. Pet Shop
2. Pet Grooming Services
3. Tinder for dogs or even owners (Dating services? LOL)
4. Dog community platform (One-stop platform to access all information to address different needs relating to their pet dog)

# Idea Prioritization Grid





# World: Data and Policy



AI Essentials Framework

# Data

HAVE	WANTED
<p><b>Public</b> From Kaggle: Dog Breed Identification - 120 dog breeds</p> <p>Included: Train(10.2k files) and Test(10.4k files) datasets</p>	<p><b>Public</b> Worldwide, the FCI lists 360 officially recognized dog breeds but we could not find that in one dataset.</p>

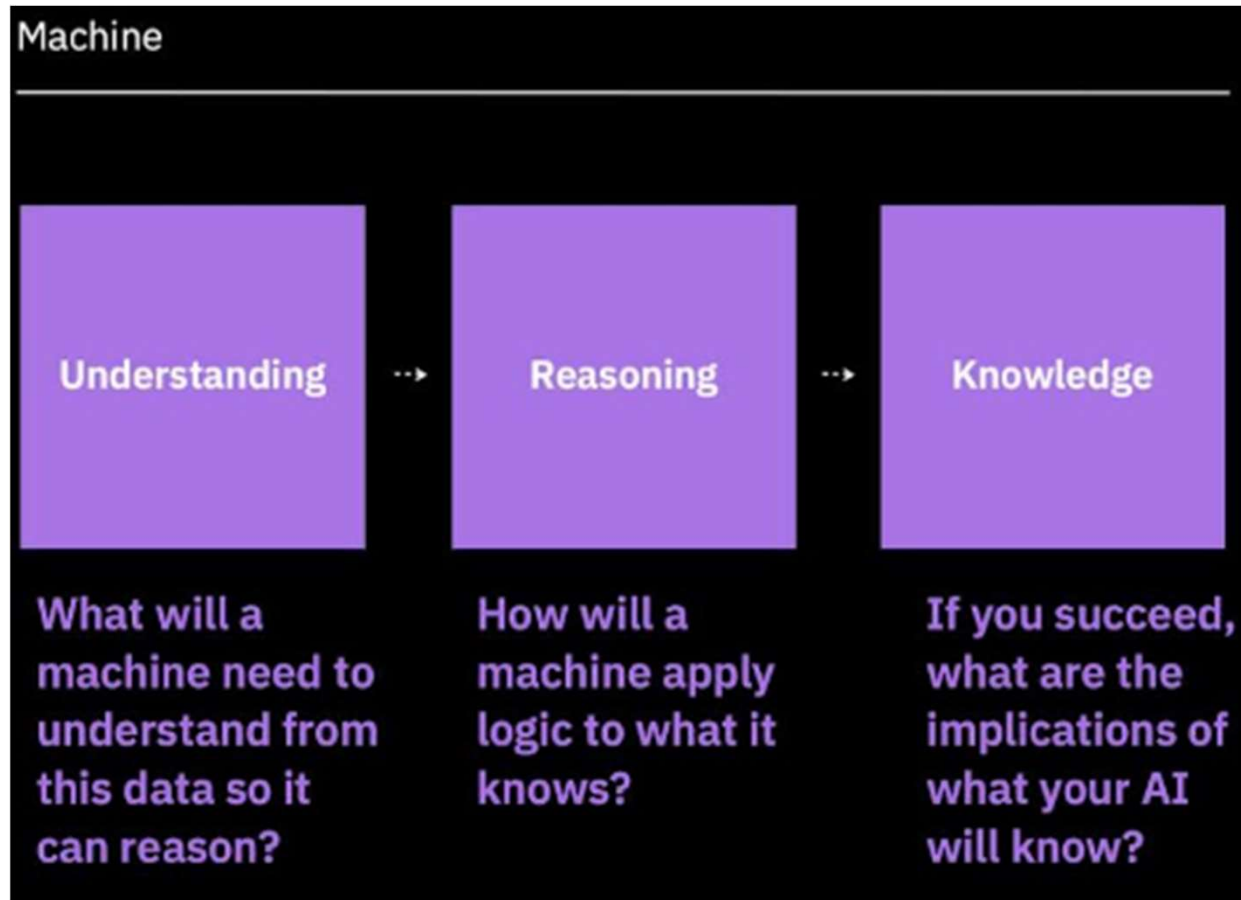
## Input (750.43 MiB)

### 📁 Data Sources

- ▼ 🐶 Dog Breed Identification
  - ▶ 📁 test
  - ▶ 📁 train
  - 📄 labels.csv
  - 📄 sample\_submission.c...

AI Essentials Framework

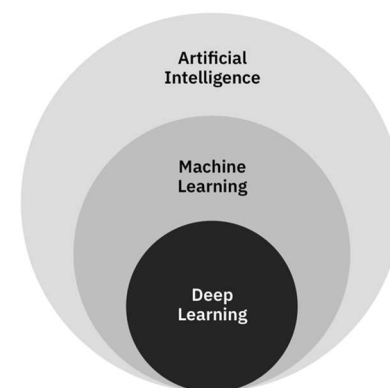
# Machine: Understanding, Reasoning and Knowledge



AI Essentials Framework

# Understanding (AI)

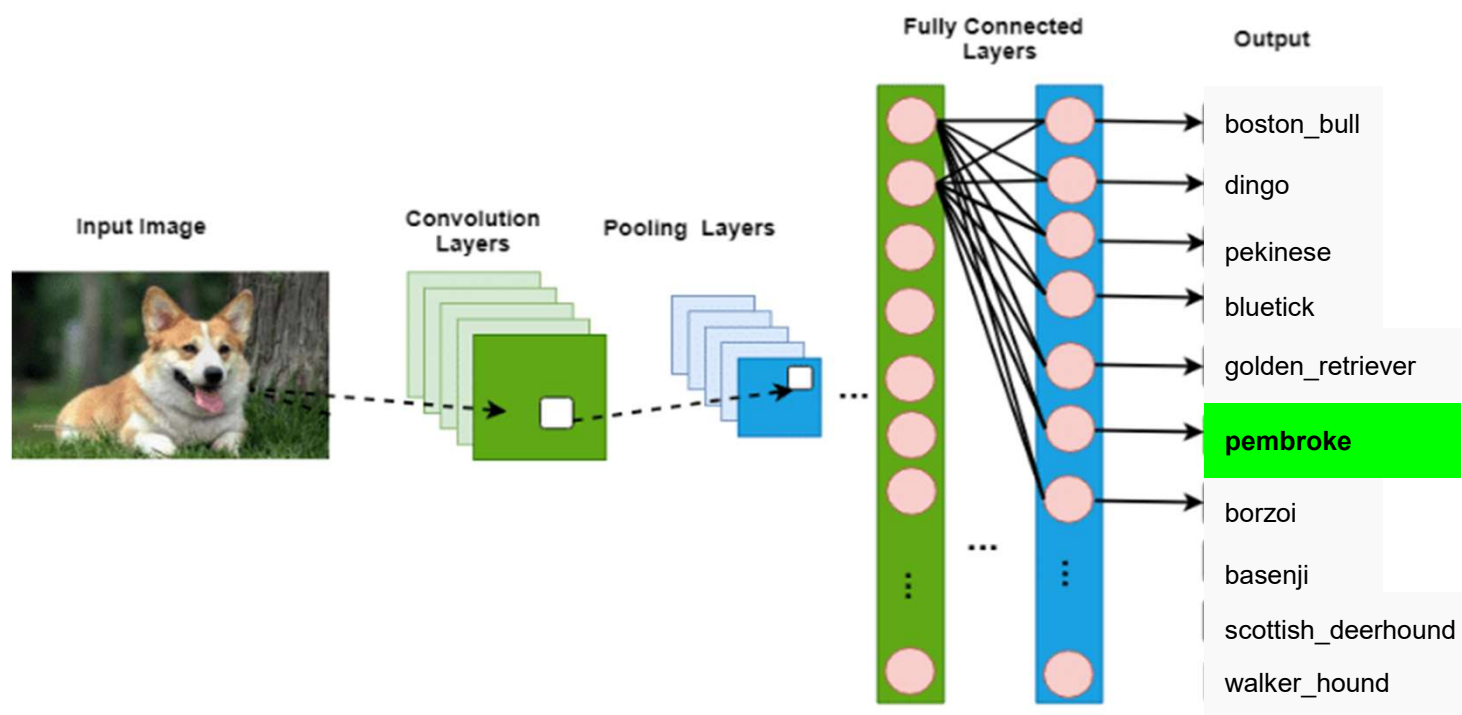
- The AI need to **identify the photo of the dog** (unstructured data) from the user input in order to classify the type of dog.
- Convolutional Neural Networks (CNN) would ideally be used.
- In **deep learning**, CNN is a class of **deep neural networks**, most commonly applied to analyze visual imagery.



# Understanding (AI)

Below is a simplified CNN architecture for dog classification from user photo input.

We need to teach the AI with a set of dog images data (for each type of dog) so that AI can identify the dog.



# Understanding (Chatbot)

- Our team decided to implement an additional AI Chatbot to interact with Dog Lovers to have a better user experience.
- AI Chatbot will be created using **IBM Watson Assistant** which is what we are familiar with.
- We also came out with ideas of adding weblink, iframe and webhook to give Dog Lovers more information.



# Reasoning

- Initially we proposed a site for dog lovers to identify their dogs but the ideas objective seems very minuscule, with dog breed identification being the only functions.
- So we propose having a pet shop with all the added services, but ideas became too big and complicated with the need to process live informations like the price of dogs, dog food and etc.





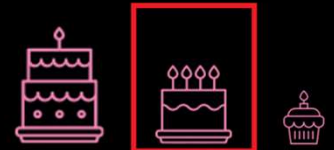
# Reasoning

- We decided on a platform for dog lovers to identify and acquire information about the dog breeds with an additional option to interact with other like-minded dog lovers at their own discretion.
- What our machine knows:  
[Identify dog breeds](#)
- What additional functionality can our machine provides:  
[Acquired information about the various dog breeds](#)
- Additional functionality that our machine cannot provides:  
[User to user interactions \(at their own discretion\)](#)

How big is your Big Idea?

Does it seem too big?

Too small?



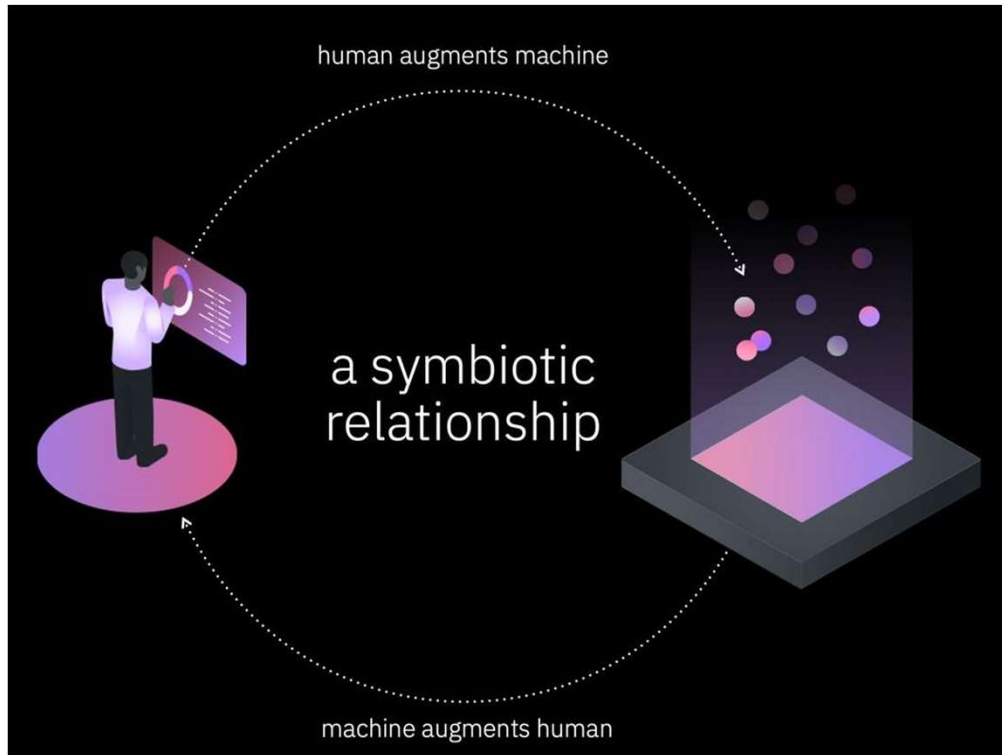
AI Essentials Framework

# Reasoning

## Reasoning Statement

Dog lovers can identify various dog breeds and gain more understanding of dogs by having a platform that provides knowledge on this topic based on a database of photos on various dogs breeds and informations that is derived from the understanding of these photos.

# Knowledge



## Reflections

Designed a relationship with Users

Build trust by empowering the users

Always not forgetting about the Intent

Why are we doing this AI?

How our AI impact people?

Don't focus on short-term outcomes

Imperative to make ethical decision at the very beginning of the AI creation process

AI Essentials Framework

# Knowledge

## 5 Focus Areas - Everyday Ethics for AI

- |                     |  |
|---------------------|--|
| 1. Accountability   | - Every person and the companies that invested in the system           |
| 2. Value Alignment  | - Align with the norms and values of the user group                    |
| 3. Explainability   | - Easily perceive, detect and understand its decision process          |
| 4. User Data Rights | - Protect user data and preserve the user's power over access and uses |
| 5. Fairness         | - Minimize bias and promote inclusiveness                              |

# Knowledge

## Layers of Effect

**Primary**  
*Intended and known*

**Secondary**  
*Intended and known*

**Tertiary**  
*Unintended and possibly known*

### Image Classification

Primary: to help anyone to identify the dog breed by uploading the photo to the app to get a result

Secondary: users may eventually also want the app to identify other animals other than dogs

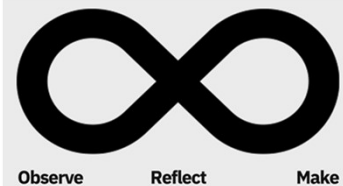
Tertiary: attrition of users after novelty of the app has worn off

### Chatbot

Primary: to bring users to the correct site to look up information based on their specific needs

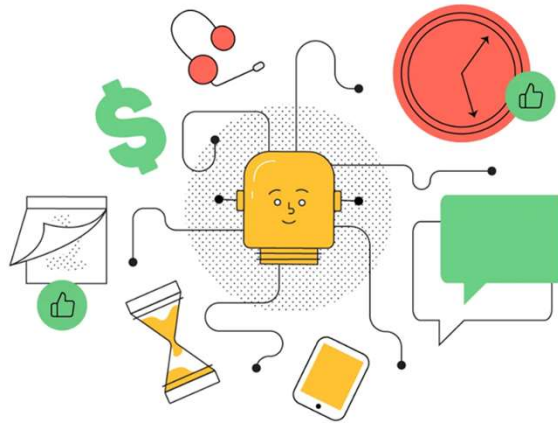
Secondary: to partner with dog/pet-related service providers to provide a more reliable one-stop service to users

Tertiary: creepy chatbot that gives 'too personal' recommendation; implicated by unethical business partners or unforeseen unfortunate circumstances



**AI Essentials Framework**

# Project Planning and Design



# Team Members Allocation

Scrum Master: Tan Wei Hong; Iris Ong (Vice-Master)

Image Recognition AI:

Data collection/cleansing: Praveen Kumar; Ng Chee Sin

Design and Deployment: Praveen Kumar

Chatbot:

Robin Lim (Lead); Iris Ong; Tan Wei Hong

Integration:

Ng Chee Sin; Chin Gon Sin; Praveen Kumar

Website:

Web developer: Chin Gon Sin

Web designer: Ng Chee Sin



Project Planning and Design



# Project Objective and Tools

## Objective:

1. Allows user to upload a photo to identify the breed of dogs through our AI from the webpage (PC and Mobile).
2. A chatbot for user to have simple interaction and to address specific queries.
3. A website (PC and Mobile) where dog lovers come to for doggie knowhow and upcoming events.

## Main Tools Used:

**Mural** - for collaboration, Enterprise Design Thinking and AI Design Essentials

**Kaggle** - collaborate with other teammates, find datasets, use GPU integrated notebooks to solve data science challenges and convert model to 'h5' file

**Python** - is a computer programming language used to build websites and software, automate tasks, and conduct data analysis.

**Github** - software can use a centralized repository where users can upload, manage code and make synchronistic edits. Files can be forked with teammates.

**Heroku** - deploy, manage and host our website

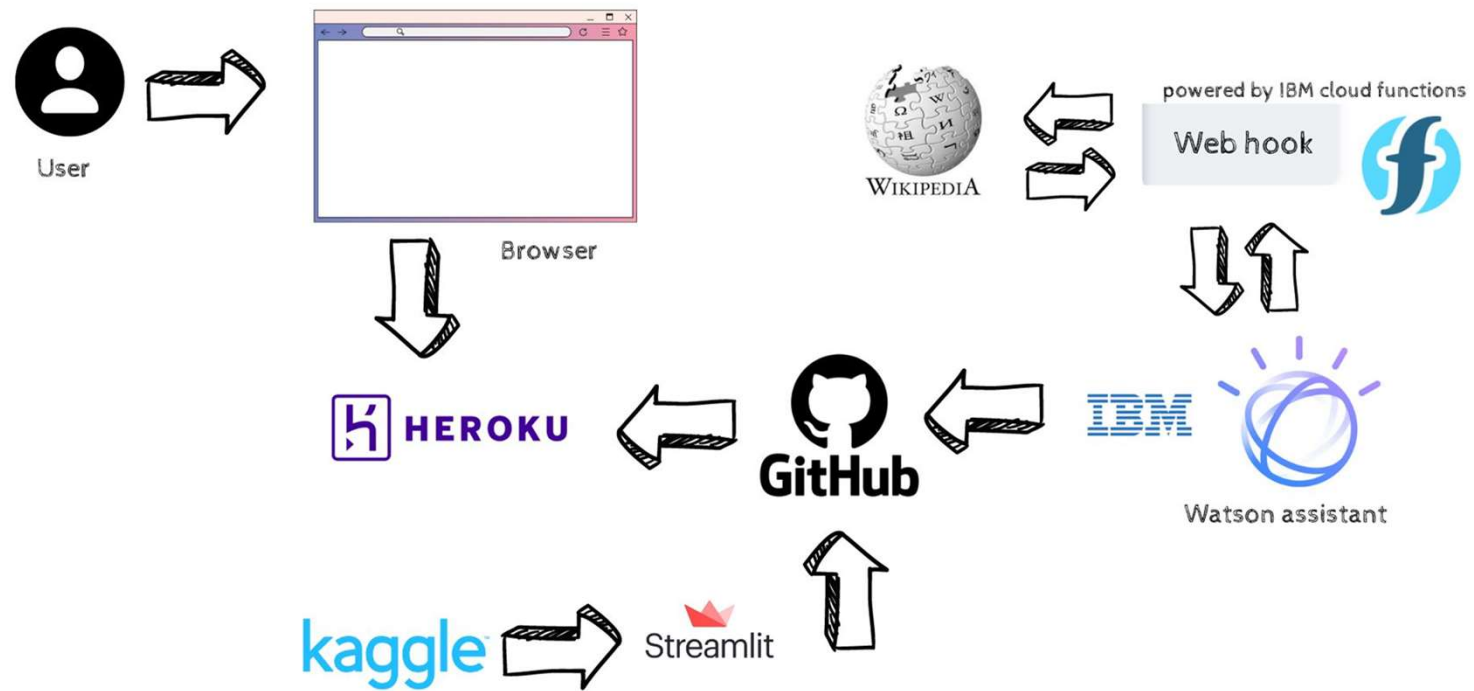
**IBM Watson Assistant** - for Chatbot with Iframe & Webhook to Wikipedia

# Technical Planning and Consideration

- 1) Use tools from IBM Cloud
  - Simple
  - Easy to implement especially Watson Assistant
  - Fast development
  
- 1) Get data from Kaggle
  - Wide variety of ready open source data
  - Using Python
  
- 1) Use known free tools (Kaggle; Github; Heroku; etc.)
  - Familiar to use as it is taught in our IBM course
  
- 1) Team members allocation - Putting our different expertise into the right places works wonders!
  
- 1) Time management - We breakdown the project and assign the different tasks to the members so that we can complete it ahead of schedule.

Project Planning and Design

## Integration/Technical Architecture



Project Planning and Design

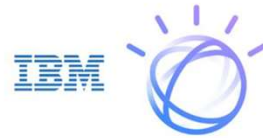
## Implementation Overview



Heroku and Streamlit as a platform to upload our website to the internet



Github as a developing platform for our website and our application.



Chatbot using watson assistant



Our web hook to wikipedia is powered by IBM cloud functions

Project Planning and Design

# Dog Identification AI



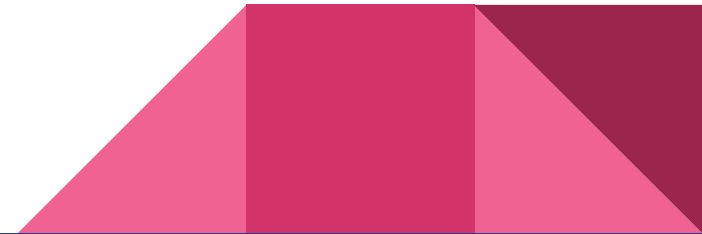
[Kaggle Link for Code](#)

## Dog Breed Classification Using a Stacked Pre-Trained Model

Detecting a Dog Breed Is Really A Huge Task. Because There Are About *Hundreds Of Breeds Of Dog* In This World And We Need To *Train Our Neural Network* Aka Deep Learning Algorithm On that Huge Data. But Fortunately Computers Now Are Very Powerful and Especially with *GPU Acceleration*, Training a Neural Network Boiled Down To A *Doable Task* and we have *Kaggle* which provides public data (Datasets) and a platform(Jupyter Notebook) to run these programs.

There are several ways to classify dog breeds, but we used the pre-trained stacked model for this purpose. In our dataset, there are 120 dog breeds to classify.

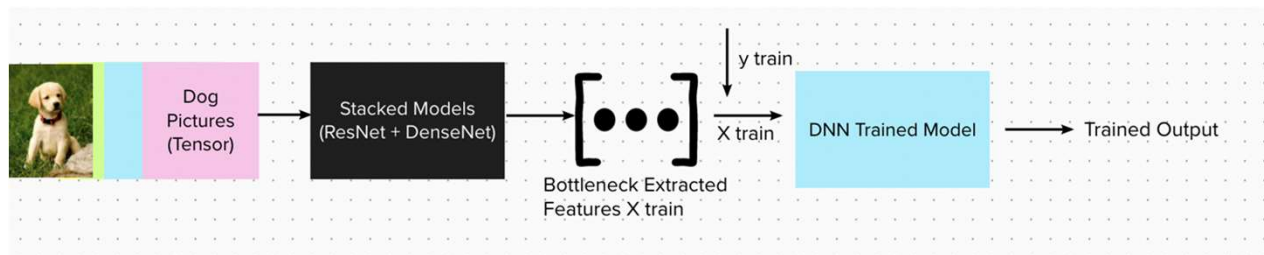
We tried to stack Resnet50V2, Densenet121 from Keras library and it gave good results after bottleneck feature extraction. These pre-trained models are trained on ImageNet open dataset.



# Workflow

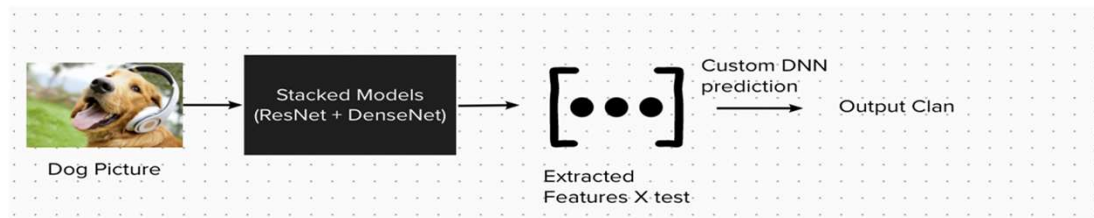
## Training

We first extracted bottleneck features using the stacked pre-trained model and then trained a simple dense neural network for the classification of extracted features on their labels.



## Testing

For the testing or prediction part, we have taken input to extract the features, and pass the extracted features as input to the simple DNN model for the final prediction. the prediction is a class number later converted that class number into their respective dog breed label.





# Model Stacking

The idea of model stacking is very simple, just concatenate the outputs of multiple models using concatenate layer

```
from keras.applications.resnet_v2 import ResNet50V2 , preprocess_input as resnet_preprocess
from keras.applications.densenet import DenseNet121, preprocess_input as densenet_preprocess
from keras.layers.merge import concatenate

input_shape = (331,331,3)
input_layer = Input(shape=input_shape)

#first extractor inception_resnet
preprocessor_resnet = Lambda(resnet_preprocess)(input_layer)
inception_resnet = ResNet50V2(weights = 'imagenet',
                                include_top = False, input_shape = input_shape, pooling = 'avg')(preprocessor_resnet)

preprocessor_densenet = Lambda(densenet_preprocess)(input_layer)
densenet = DenseNet121(weights = 'imagenet',
                       include_top = False, input_shape = input_shape, pooling = 'avg')(preprocessor_densenet)

merge = concatenate([inception_resnet, densenet])
model = Model(inputs = input_layer, outputs = merge)
```

# Model Summary

```
model.summary()
```

Model: "model"

Layer (type)	Output Shape	Param #	Connected to
=====			
input_1 (InputLayer)	[(None, 331, 331, 3)]	0	
lambda (Lambda)	(None, 331, 331, 3)	0	input_1[0][0]
lambda_1 (Lambda)	(None, 331, 331, 3)	0	input_1[0][0]
resnet50v2 (Functional)	(None, 2048)	23564800	lambda[0][0]
densenet121 (Functional)	(None, 1024)	7037504	lambda_1[0][0]
concatenate (Concatenate)	(None, 3072)	0	resnet50v2[0][0] densenet121[0][0]
=====			

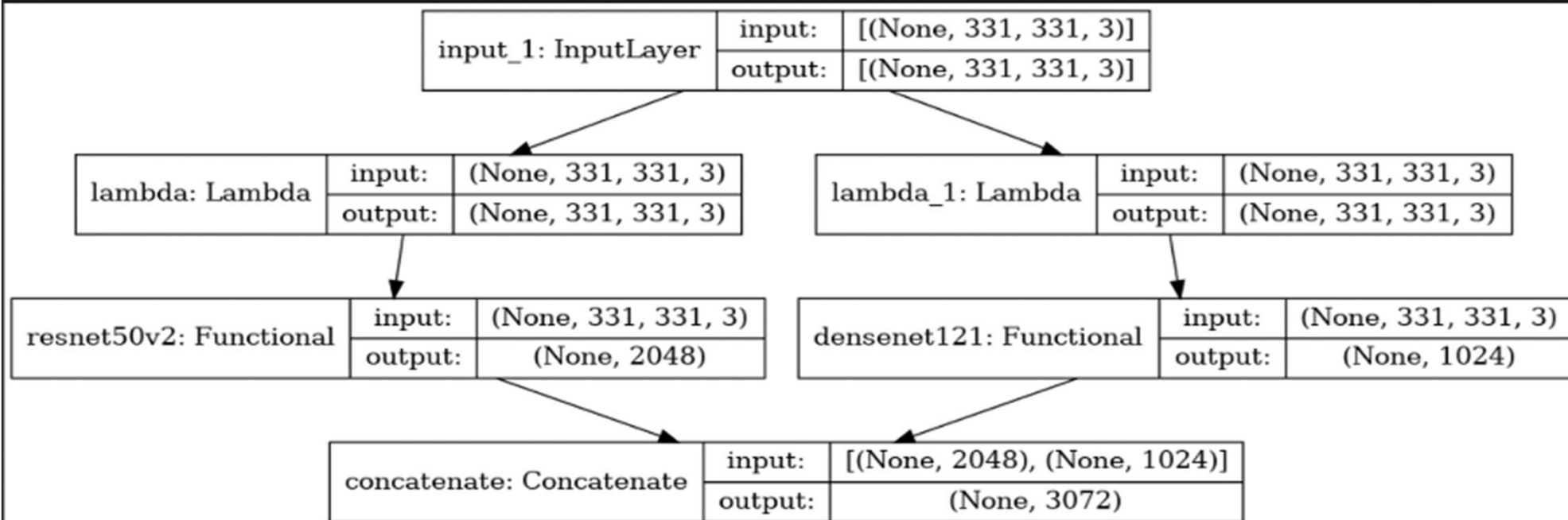
Total params: 30,602,304

Trainable params: 30,473,216

Non-trainable params: 129,088

# Architecture

```
from tensorflow.keras.utils import plot_model
plot_model(model, show_shapes = True)
```



# Implementation

## Loading the Data

Creating a data frame `labels_dataframe` containing dog breed class numbers with the path of images. Here we have mapped every dog breed label with some numbers

```
#Data Paths
train_dir = '/kaggle/input/dog-breed-identification/train/'
test_dir = '/kaggle/input/dog-breed-identification/test/'
#Count/Print train and test samples.

#Read train labels.
labels_dataframe = pd.read_csv('/kaggle/input/dog-breed-identification/labels.csv')
#Read sample_submission file to be modified by predicted labels.
sample_df = pd.read_csv('/kaggle/input/dog-breed-identification/sample_submission.csv')
#Inspect labels_dataframe.
labels_dataframe.head(5)|
# here sample df is the format of data to be submitted into kaggle
```

```
y = to_categorical(labels_dataframe.breed)
# encoded our y variable to pass in our model
```

# Extract bottleneck features

This step is very crucial and time-consuming, here we extracted features of all images using the stacked model. Extracted in batches in order to avoid RAM insufficiency problems. Extracted features is our `X_train` and class `Id` is our `y_train`. `X_train` contains all the extracted features which are input for the final predictor model.

```
# for feature_extraction dataframe must have to contain file_name and breed columns
def feature_extractor(df):
    img_size = (331,331,3)
    data_size = len(df)
    batch_size = 20
    X = np.zeros([data_size,3072], dtype=np.uint8)
    # y = np.zeros([data_size,120], dtype=np.uint8)
    datagen = ImageDataGenerator() # here we dont need to do any image augmentation because we are prediction features
    generator = datagen.flow_from_dataframe(df,
    x_col = 'file_name', class_mode = None,
    batch_size=20, shuffle = False, target_size = (img_size[:2]), color_mode = 'rgb')
    i = 0

    for input_batch in tqdm(generator):
        input_batch = model.predict(input_batch)
        X[i * batch_size : (i + 1) * batch_size] = input_batch
        i += 1
        if i * batch_size >= data_size:
            break
    return X
```

```
X = feature_extractor(labels_dataframe)
```

Found 10222 validated image filenames.

# Final Predictor Model

```
dnn = keras.models.Sequential([
    InputLayer(X.shape[1:]),
    Dropout(0.7),
    Dense(n_classes, activation='softmax')
])

dnn.compile(optimizer='adam',
            loss='categorical_crossentropy',
            metrics=['accuracy'])
```

For efficient training, we can make use of callbacks.

```
from keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLRonPlateau
#Prepare call backs
EarlyStop_callback = keras.callbacks.EarlyStopping(monitor='val_loss', patience=20, restore_best_weights=True)
checkpoint = ModelCheckpoint('/kaggle/working/checkpoint',
                             monitor = 'val_loss', mode = 'min', save_best_only= True)
lr = ReduceLRonPlateau(monitor = 'val_loss', factor = 0.5, patience = 3, min_lr = 0.00001)
my_callback=[EarlyStop_callback, checkpoint]
```

# Training Phase

Training the final\_model on X\_train and y\_train and our X\_test ,y\_test with the help of validation\_split .Using validation\_split = 0.1 , the dataset will be split into (90%) and (10%).

```
#Train simple DNN on extracted features.
h = dnn.fit(X , y,
            batch_size=128,
            epochs=60,
            validation_split=0.1 ,
            callbacks = my_callback)
```

Epoch 1/60  
72/72 [=====] - 1s 5ms/step - loss: 3.2729 - accuracy: 0.2799 - val\_loss: 1.4906 - val\_accuracy: 0.7576  
2022-07-19 03:47:42.040221: W tensorflow/python/util/util.cc:348] Sets are not currently considered sequences, but this may change in the future, so consider avoiding using them.

Epoch 2/60  
72/72 [=====] - 0s 3ms/step - loss: 1.3320 - accuracy: 0.6947 - val\_loss: 0.8827 - val\_accuracy: 0.8280

Epoch 3/60  
72/72 [=====] - 0s 3ms/step - loss: 0.8870 - accuracy: 0.7889 - val\_loss: 0.6989 - val\_accuracy: 0.8397

Epoch 4/60  
72/72 [=====] - 0s 3ms/step - loss: 0.6915 - accuracy: 0.8343 - val\_loss: 0.6149 - val\_accuracy: 0.8495

Epoch 5/60  
72/72 [=====] - 0s 3ms/step - loss: 0.5726 - accuracy: 0.8592 - val\_loss: 0.5649 - val\_accuracy: 0.8592

Epoch 6/60  
72/72 [=====] - 0s 3ms/step - loss: 0.4983 - accuracy: 0.8754 - val\_loss: 0.5389 - val\_accuracy: 0.8553

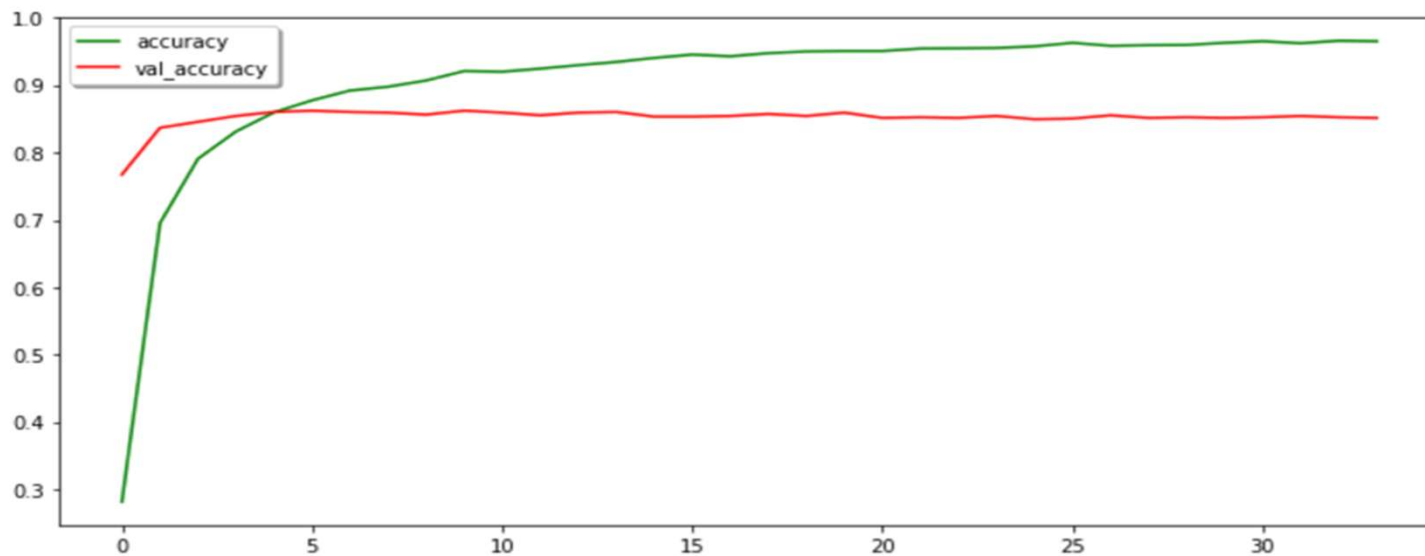
Epoch 7/60  
72/72 [=====] - 0s 3ms/step - loss: 0.4411 - accuracy: 0.8905 - val\_loss: 0.5139 - val\_accuracy: 0.8631

Epoch 8/60  
72/72 [=====] - 0s 3ms/step - loss: 0.3919 - accuracy: 0.9018 - val\_loss: 0.5020 - val\_accuracy: 0.8631

Epoch 9/60

## Plot the Result

Plotting the training history gives us an idea about the performance and training. here 'history\_graph' is the history object that contains all the training graphs for every epoch. Finally, we have trained our model and got a **validation accuracy of 85%**, this is a good start. though it's not an end, we can further improve this accuracy by using some fine-tuning.





# Model Saving

Saving the trained model for further development. Using the saved model we can host the model on a website, can create an app that will be able to classify dog breeds.

```
# saving the model
from keras.models import load_model
dnn.save('/kaggle/working/dogbreed.h5')
```

[Link for Github files](#)



# Model Deployment

We used streamlit and heroku for deploying our model

Loading the saved model : In this code chunk, we are loading the different categories of Dog Breeds using the pickle file and then we are loading the weights file (.h5 file) that has training weights.

```
2  import os
3  import numpy as np
4  import pickle
5  import tensorflow as tf
6
7  from tensorflow.keras import layers
8  from tensorflow.keras import models,utils
9  import pandas as pd
10 from tensorflow.keras.models import load_model
11 from tensorflow.keras.preprocessing.image import load_img,img_to_array
12 from tensorflow.python.keras import utils
13 current_path = os.getcwd()
14 dog_breeds_category_path = os.path.join(current_path, 'static/dog_breeds_category.pickle')
15 predictor_model = load_model(r'static/dogbreed.h5')
16 with open(dog_breeds_category_path, 'rb') as handle:
17     dog_breeds = pickle.load(handle)
18
```

# Model Deployment

Now the predictor function is defined that takes the image path as input and returns prediction.

```
def predictor(img_path): # here image is file name
    # base_path = os.path.join(current_path, 'static\images\cache')
    # path = os.path.join(base_path, image_name)
    img = load_img(img_path, target_size=(331,331))
    # print(path)
    # img = cv2.resize(img, (331,331))
    img = img_to_array(img)
    img = np.expand_dims(img, axis = 0)
    features = feature_extractor.predict(img)
    prediction = predictor_model.predict(features)*100
    prediction = pd.DataFrame(np.round(prediction,1), columns = dog_breeds).transpose()
    prediction.columns = ['values']
    prediction = prediction.nlargest(5, 'values')
    prediction = prediction.reset_index()
    prediction.columns = ['name', 'values']
    return(prediction)
```

## Creating Frontend:

r function

```
import streamlit as st
import os
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_theme(style="darkgrid")
sns.set()

from PIL import Image
from helper import *
st.title('Dog Breed Classifier')

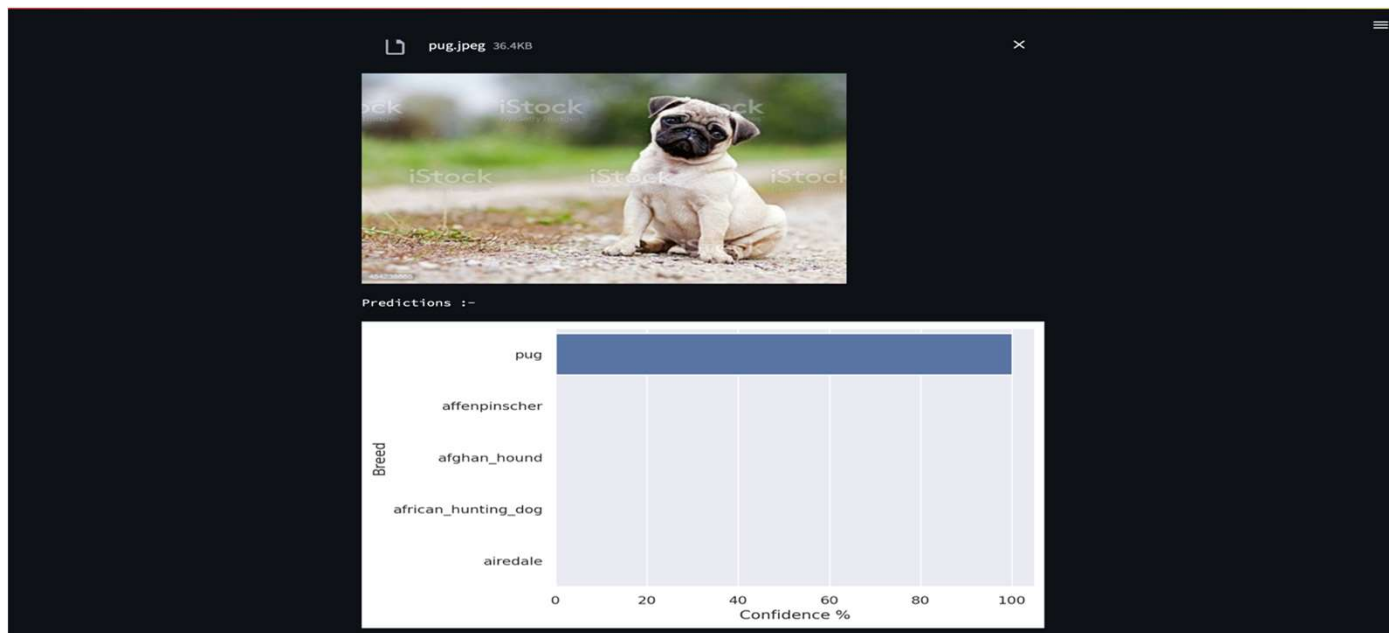
def save_uploaded_file(uploaded_file):
    try:
        with open(os.path.join('uploaded', uploaded_file.name), 'wb') as f:
            f.write(uploaded_file.getbuffer())
        return 1
    except:
        return 0

uploaded_file = st.file_uploader("Upload Image")
if uploaded_file is not None:
    if save_uploaded_file(uploaded_file):
        # display the file
        display_image = Image.open(uploaded_file)
        display_image = display_image.resize((500, 300))
        st.image(display_image)
        prediction = predictor(os.path.join('uploaded', uploaded_file.name))
        print(prediction)
        os.remove('uploaded/'+uploaded_file.name)
        # drawing graphs
        st.text('Predictions :-')
        fig, ax = plt.subplots()
        ax = sns.barplot(y='name', x='values', data=prediction, order=prediction.sort_values('values', ascending=False).name)
        ax.set(xlabel='Confidence %', ylabel='Breed')

        st.pyplot(fig)
```

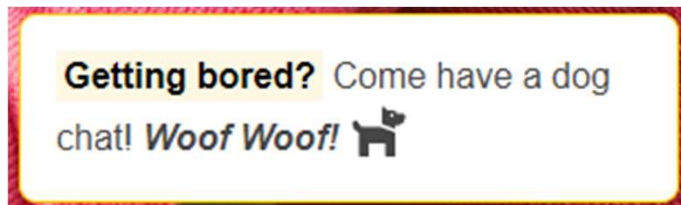
# Final look of Classification App on Heroku

Heroku is a free online model hosting service. you can host the model on the cloud for free. Deploying apps on Heroku is more flexible, Managing your app, package versions, storage is a lot easier with Heroku. [Link for the Classification App on Heroku](#)



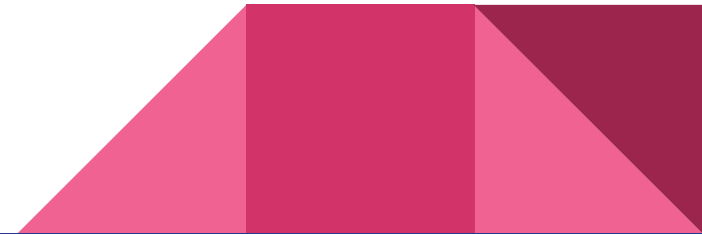
# Chatbot

Conversational Chatbot

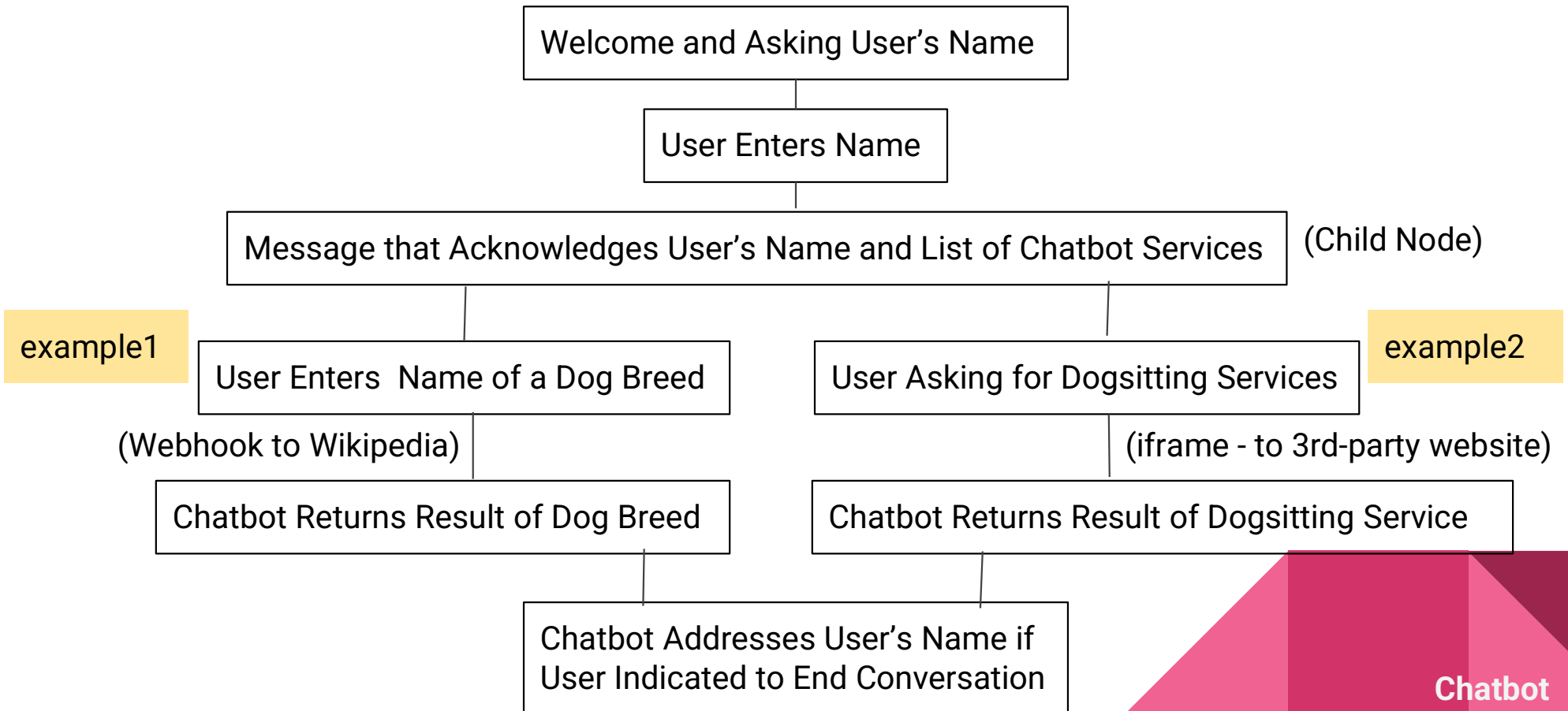


## Why Chatbot?


- Value-add to our website
- 24/7 support to users
- Prefer texting and self-service
- Increased user engagement
- Continued trust and reliability



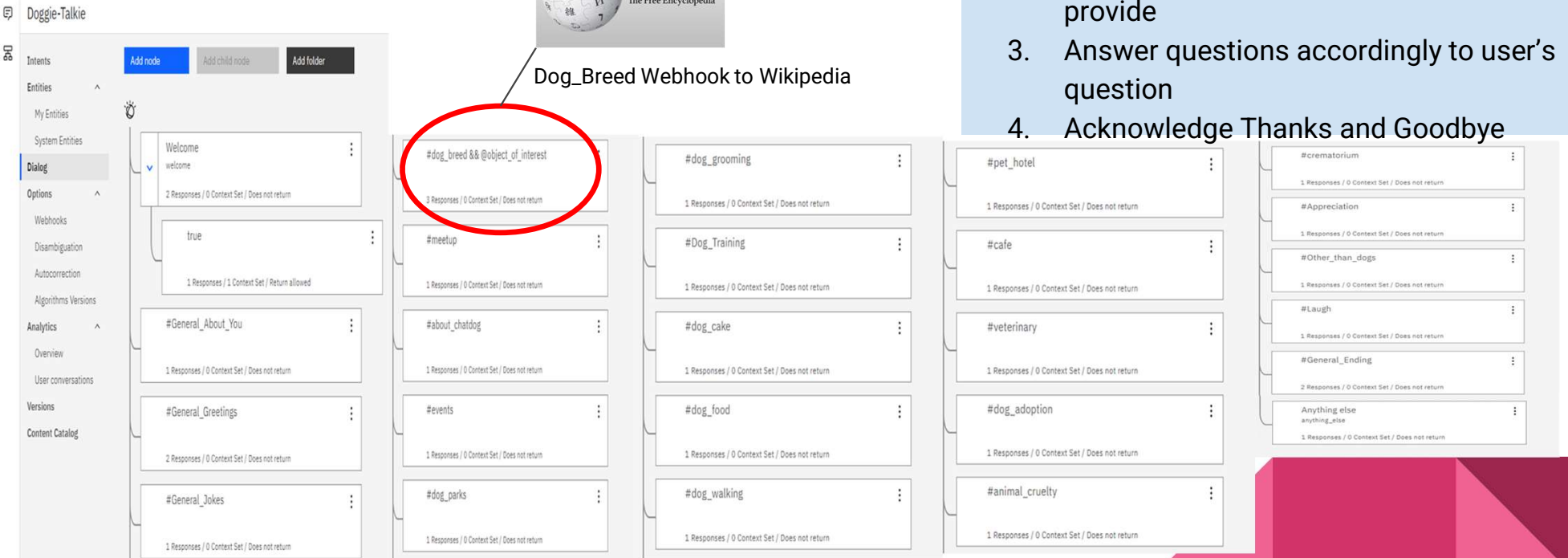
# Flow Conversation Design



# Watson Assistance



Dog\_Breed Webhook to Wikipedia



The screenshot shows the Watson Assistant interface for a chatbot named 'Doggie-Talkie'. The left sidebar contains navigation options: Intents, Entities, My Entities, System Entities, Dialog, Options, Webhooks, Disambiguation, Autocorrection, Algorithms Versions, Analytics, Overview, User conversations, Versions, and Content Catalog. The main area displays a dialog flow with several nodes. The node labeled '#dog\_breed && @object\_of\_interest' is circled in red. Other nodes include 'Welcome', 'true', '#General\_About\_You', '#General\_Greetings', '#General\_Jokes', '#dog\_grooming', '#Dog\_Training', '#dog\_cake', '#dog\_food', '#dog\_walking', '#pet\_hotel', '#veterinary', '#dog\_adoption', '#animal\_cruelty', '#crematorium', '#Appreciation', '#Other\_than\_dogs', '#Laugh', '#General\_Ending', and 'Anything else anything\_else'. Each node shows its response count and context set status.

1. Welcome and asking for the user name
2. Address the user's name and provide a list of information that the chatbot can provide
3. Answer questions accordingly to user's question
4. Acknowledge Thanks and Goodbye

Chatbot



# Watson Assistant - Intent & Entities

← | #dog\_breed

User example

Type a user example here

Add unique examples of what the user might say. (Pro tip: Add at least 5 unique examples to help Watson understand)

Add example

☐ User examples (15) ↑

☐ beagle

☐ border\_collie

☐ boxer

☐ bulldog

☐ chihuahua

☐ dachshund

← | @object\_of\_interest

Dictionary (15)

Annotation (13)

☐ Values (15) ↑

Type

☐ beagle

Synonyms

☐ border\_collie

Synonyms

☐ boxer

Synonyms

☐ breed

Synonyms

☐ bulldog

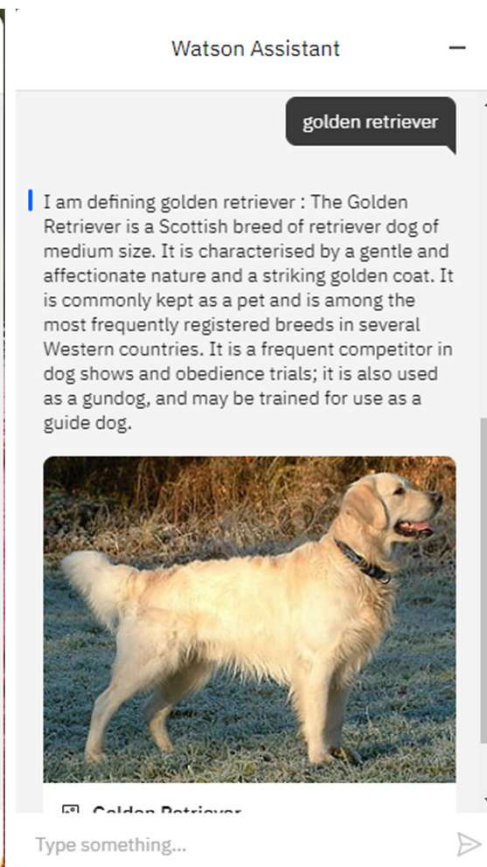
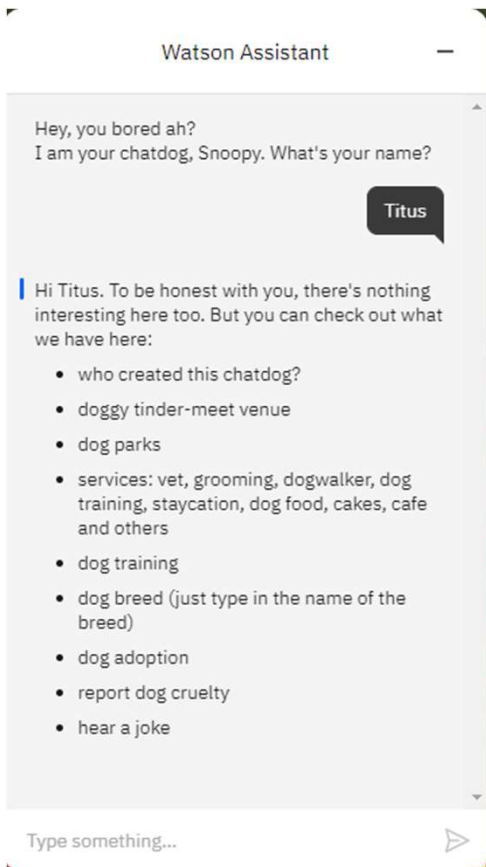
Synonyms

☐ chihuahua

Synonyms

Chatbot

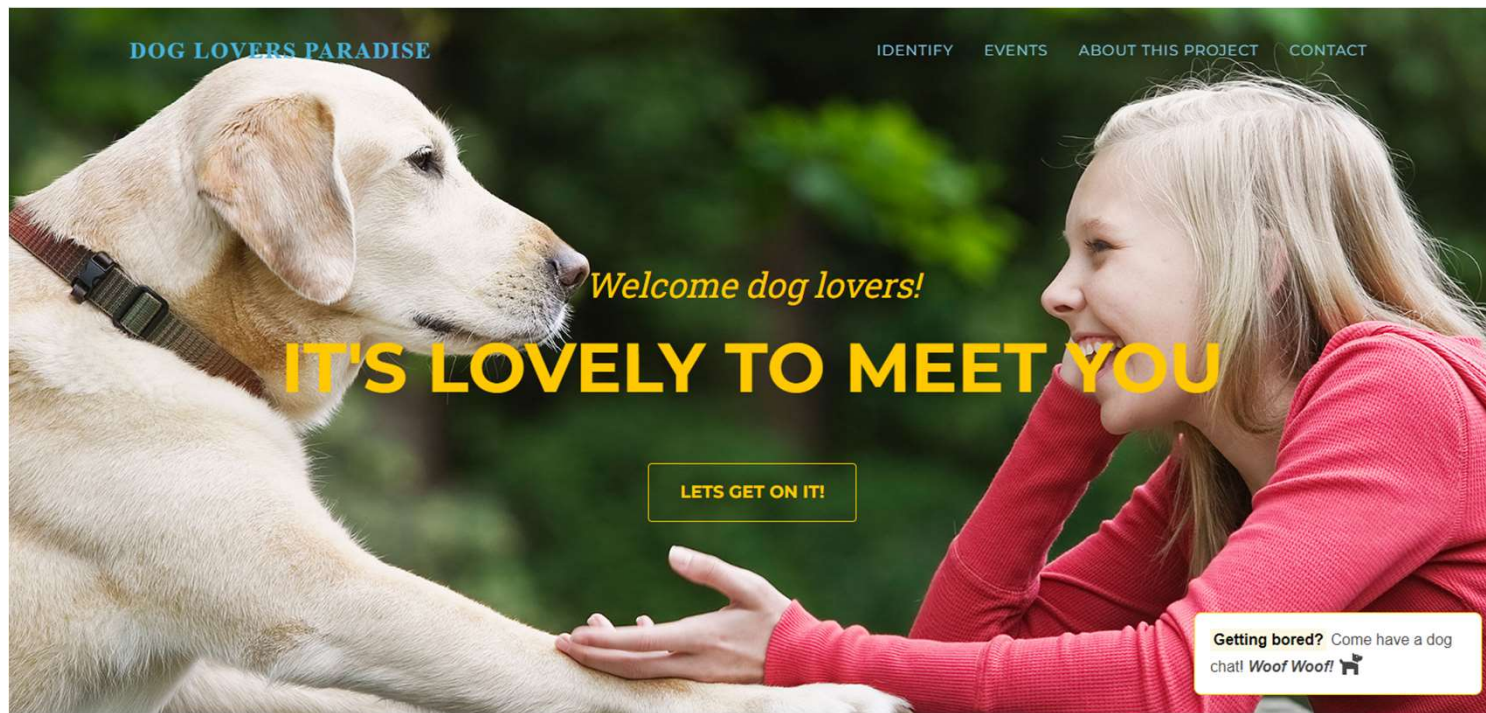
# Chatbot - Screenshot



- Personalisation
- Webhook to Wikipedia
- Using iframe for Better User Experience

Chatbot

# Project Demo



# Project Learnings/Challenges

## Constraints and Challenges

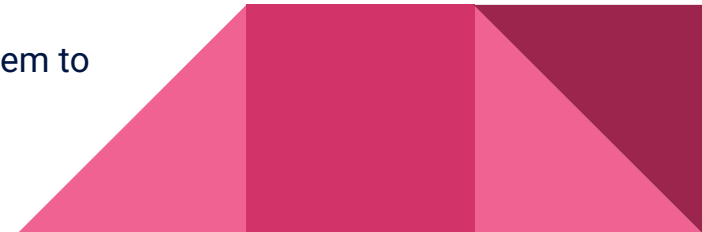
### 1. **Size of Data set**

- Multi-classification of 120 classes was chosen instead of binary classification. It was a challenge as our limited experience with binary classification already went past the data limit for deployment.
- The file size limitation free GitHub repository is a challenge. Learn a lot about website designing and integration through my teammates and googling.

### 1. **Chatbot Webhook** - Limited by Watson Assistant 30 days trial and wanted to give it a pass, Damien did not give up and found a link that guided us through the process, coupled with Iris' eye for details and innate ability to overcome obstacles, the webhook to Wikipedia finally worked.

### 1. **Chatbot** - Creating a standard, goody two-shoes chatbot is very simple. But creating a chatbot with character and personality is hard.

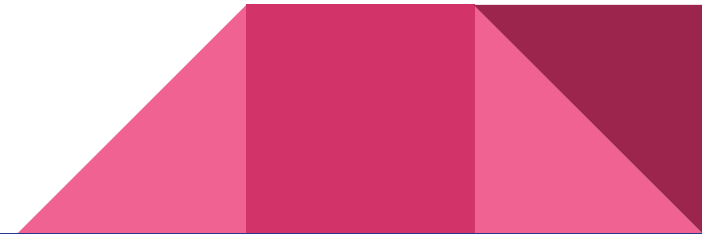
### 1. **Watson Assistant** - Duplicate keywords in different Intents caused the system to return wrong response



# Project Learnings/Challenges

## Integration and Deployment

1. Model Size - Went beyond the 25MB to upload at Github
  - a. What did we do?
    - i. Try changing the program, epoch and node layer (very time consuming)
    - ii. Extracted only weights for Model 'h5' file
2. Faced issue integrating the h5 file with html. The html container would need to be able to show the output that is generated from the 'h5' file.
  - a. What did we do?
    - i. We made a html link from our main website to our image classification website



# Team Reflections

A BIG THANK YOU to SUHAIMI. This project would not have gone this far without him pushing our boundaries to learn beyond the intended scope in AI Prac. ❤️

Damien - It is very important to maintain continuous learning throughout , you got to keep googling and try out the solutions , for example how to link up wikipedia to our chatbot. I think that communication is the key to success in a team environment. Always seek to clarify and converge when you have different ideas in mind.

Gon Sin - Everyone has their own expertise and we contribute according to what we are good at. It is always enjoyable working with people from different background and expertise.

Chee Sin - It is very tiring but fulfilling to see the final outcome, especially the process.

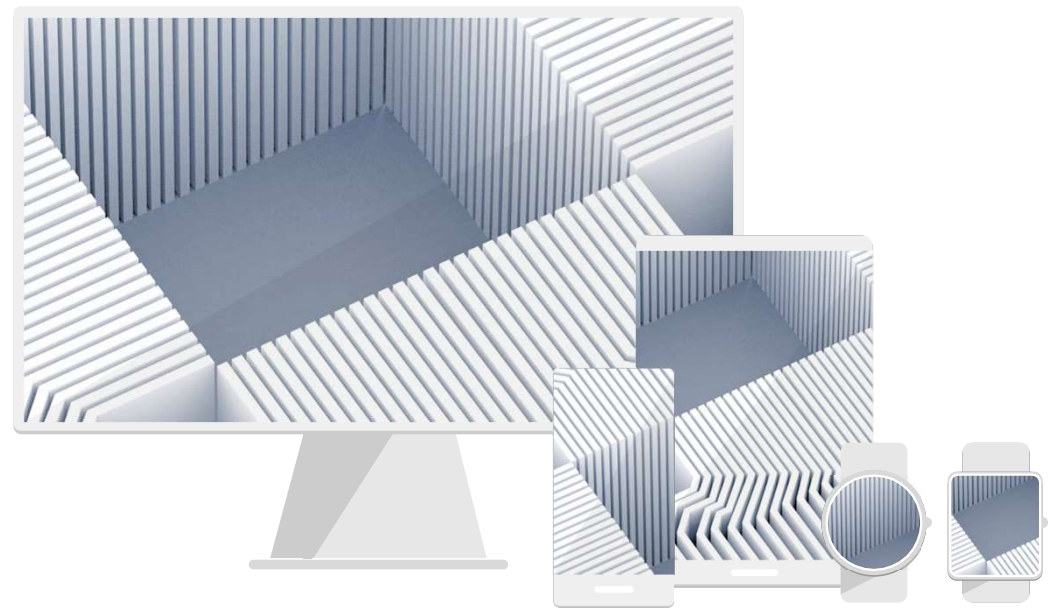
Praveen - Trying to execute the program n number of times and modifying everytime to achieve desired results was hard yet a fulfilling and great learning experience.

Iris - Looking at our end product, am sure you are jealous not to be in our team! Am thankful for a great and wonderful teammates. With such great team, no mountain is too tall to scale. ❤️

Robin - Very thankful that my teammates are geniuses and hard workers too who put in long hours through day and night to make this work.



***Thank you!***



# Credits

<https://www.techtarget.com/whatis/definition/IFrame-Inline-Frame>

<https://github.com/IBM/IBMDeveloper-recipes/blob/main/connect-watson-assistant-with-wikipedia-api-via-cloud-functions/index.md>

<https://thesmartlocal.com/read/dog-runs/>

<https://waggie.com.sg/our-services/dog-shows-and-events/>

[Gerry | Expert | Kaggle](#)

[Bootstrap · The most popular HTML, CSS, and JS library in the world. \(getbootstrap.com\)](#)

<https://funkidsjokes.com/peanuts-jokes-with-charlie-brown/>