

Fertilizers Recommendation System For Disease Prediction

☀ 1. Introduction:

✱ 1.1 Overview:

Agriculture is the most important sector in today's life. Most plants are affected by a wide variety of bacterial and fungal diseases. Diseases on plants placed a major constraint on the production and a major threat to food security. Hence, early and accurate identification of plant diseases is essential to ensure high quantity and best quality. In recent years, the number of diseases on plants and the degree of harm caused has increased due to the variation in pathogen varieties, changes in cultivation methods, and inadequate plant protection techniques.

An automated system is introduced to identify different diseases on plants by checking the symptoms shown on the leaves of the plant. Deep learning techniques are used to identify the diseases and suggest the precautions that can be taken for those diseases.

✱ 1.2 Purpose:

To Detect and recognize the plant diseases and to **recommend fertilizer**, it is necessary to provide symptoms in identifying the

disease at its earliest. Hence the authors proposed and implemented new **fertilizers Recommendation System** for crop **disease prediction**.

✳ 2.LITERATURE SURVEY:

✳ 2.1 Existing problem:

Adequate mineral nutrition is central to crop production. However, it can also exert considerable Influence on disease development. Fertilizer application can increase or decrease development of diseases caused by different pathogens, and the mechanisms responsible are complex, including effects of nutrients on plant growth, plant resistance mechanisms and direct effects on the pathogen. The effects of mineral nutrition on plant disease and the mechanisms responsible for those effects have been dealt with comprehensively elsewhere. In India, around 40% of land is kept and grown using reliable irrigation technologies, while the rest relies on the monsoon environment for water. Irrigation decreases reliance on the monsoon, increases food security, and boosts agricultural production.

Most research articles use humidity, moisture, and temperature sensors near the plant's root, with an external device handling all of the data provided by the sensors and transmitting it directly to an external display or an Android application. The application was created to measure the approximate values of temperature, humidity, and moisture sensors that were programmed into a microcontroller to manage the amount of water.

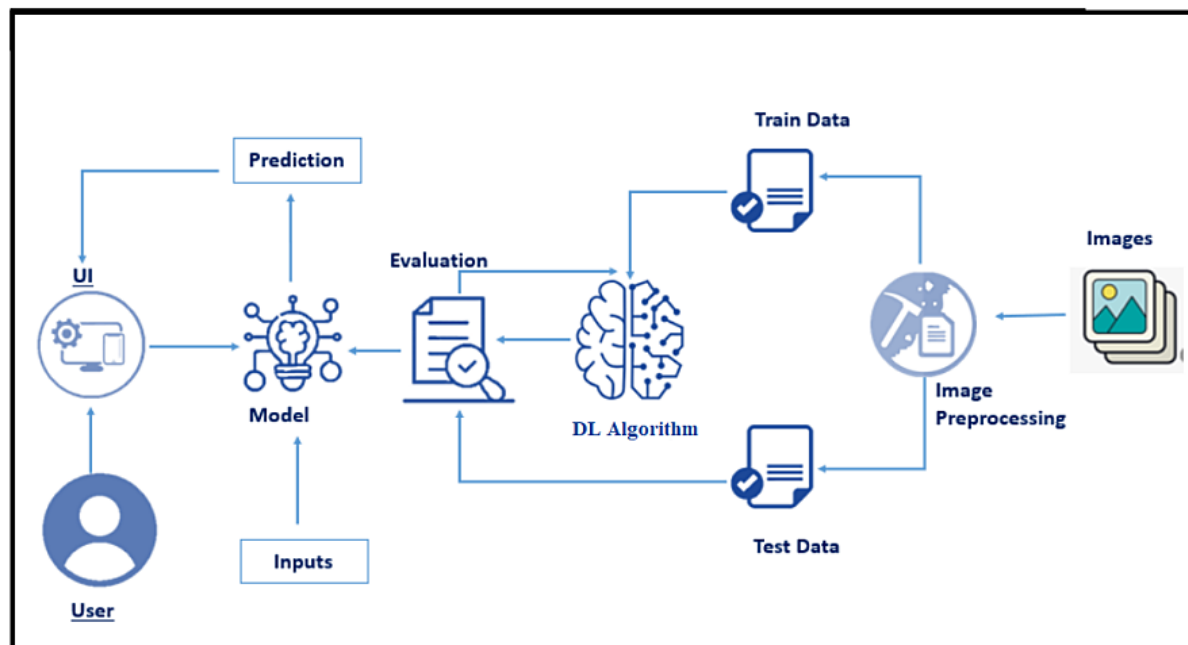
✳ 2.2 Proposed solution:

Web Application si built where :

- | |
|---|
| 1. Farmers interact with the portal build |
| 2. Interacts with the user interface to upload images of diseased leaf |
| 3. Our model built analyses the Disease and suggests the farmer with fertilizers are to be used |

✳ 3 THEORITICAL ANALYSIS :

3.1 Block diagram :



✳ 3.2 Hardware / Software designing:

To complete this project you should have the following software and packages.

Softwares:

- ☒ Anaconda Navigator
- ☒ py charm
- ☒ Visual studio code
- ☒ Jupiter notebook
- ☒ IBM watson studio

Packages:

- ☒ Tensor flow
- ☒ Keras
- ☒ Flask
- ☒ numpy
- ☒ Pandas

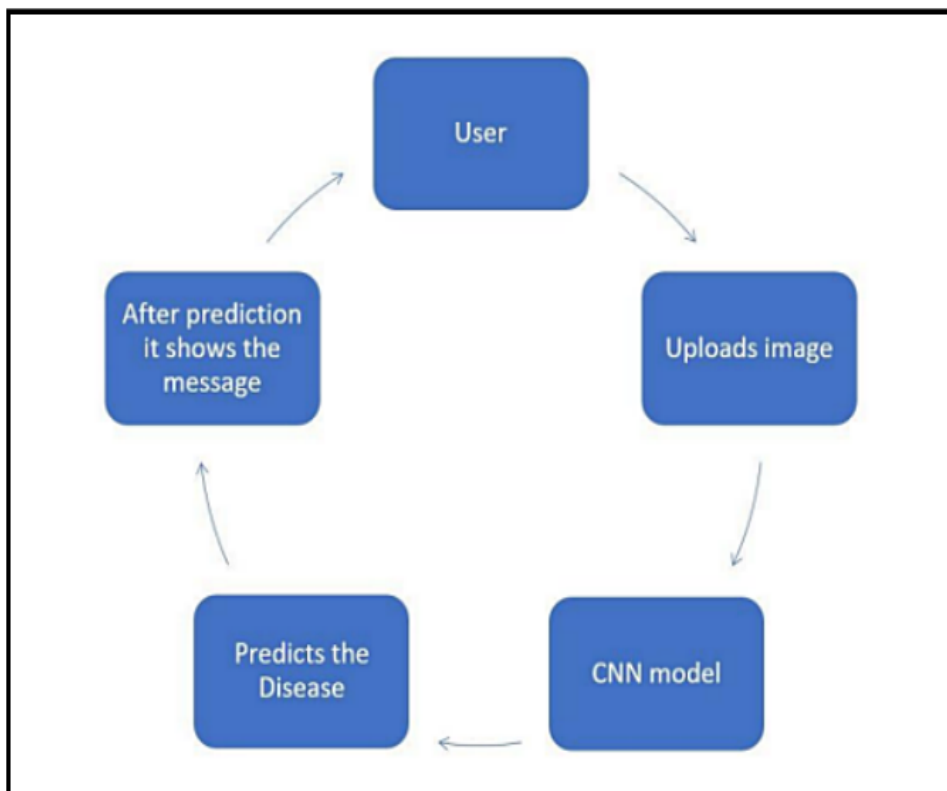
By using the above listed softwares and packages ,we build this application to take the input (image) from the farmer and detects whether the plant is infected or not. Here we use Deep learning techniques and give the out put to the user (Farmer).

✳ 4 EXPERIMENTAL INVESTIGATIONS :

Agriculture is the most important sector in today's life. Most plants are affected by a wide variety of bacterial and fungal diseases. Diseases on plants placed a major constraint on the production and a major threat to food security. Hence, early and accurate identification of plant diseases is essential to ensure high quantity and best quality. In recent years, the number of diseases on plants and the degree of harm caused has increased due to the variation in pathogen varieties, changes in cultivation methods, and inadequate plant protection techniques.

An automated system is introduced to identify different diseases on plants by checking the symptoms shown on the leaves of the plant. Deep learning techniques are used to identify the diseases and suggest the precautions that can be taken for those diseases. Analysis or the investigation made while working on the solution.

✳ 5 FLOWCHART :



To accomplish the above task you must complete the below activities and tasks :

- Download the dataset.

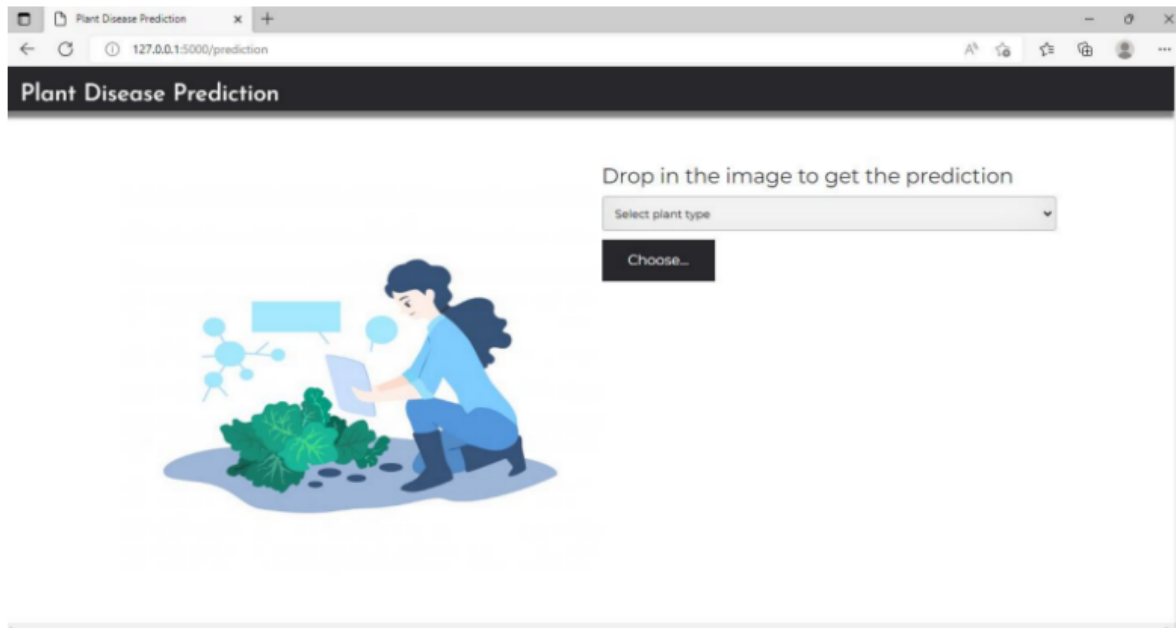
- Classify the dataset into train and test sets.
- Add the neural network layers.
- Load the trained images and fit the model.
- Test the model.
- Save the model and its dependencies.
- Build a Web application using a flask that integrates with the model built.

✳ 6 RESULT:

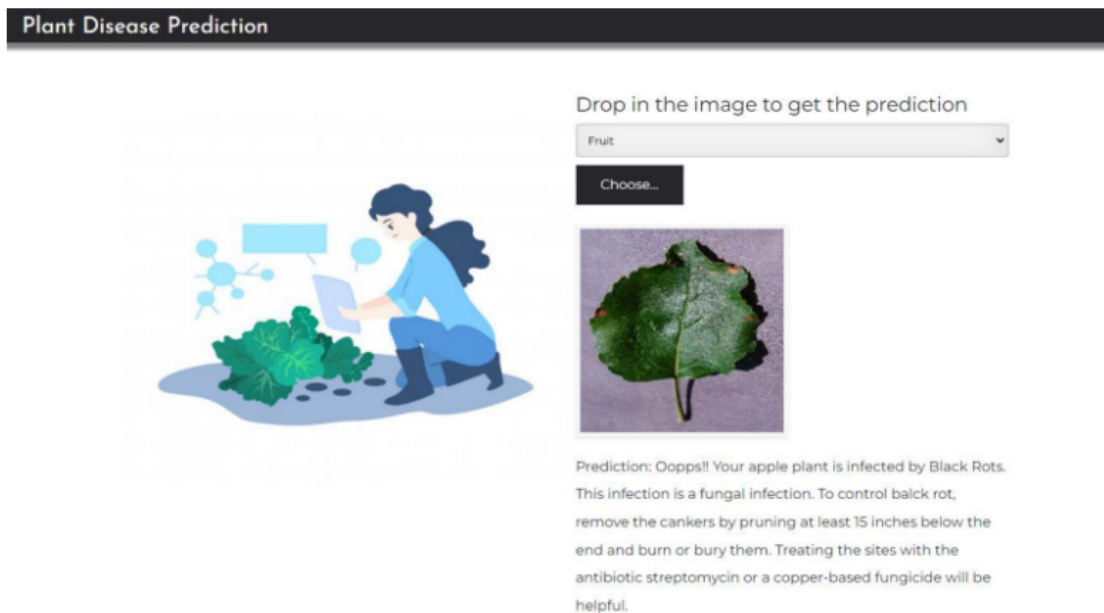
➤ **Home Page :**



➤ **Prediction Page:**



➤ **Result Page:**



Plant Disease Prediction



Drop in the image to get the prediction

Vegetable

Choose...



Prediction: Yaayy!! Your pepper plant is healthy. But, take the necessary precautions like, putting the plant where it gets at least 10 hours of direct sunlight. Keep soil evenly moist for good growth. Peppers need well draining soil that is rich and loamy, but avoid too much nitrogen in the soil. Too much nitrogen can cause plenty of leaves and little to no peppers. Your soil should have a pH between 6.0 and 6.5.

☀ 7 ADVANTAGES & DISADVANTAGES:

➤ ADVANTAGES:

- The proposed model could predict the disease just from the image of a particular plant
- Easy to use UI
- Model has some good accuracy in detecting the plant just by taking the input(leaf).

➤ DISADVANTAGES:

- Prediction is limited to few plants as we havent trained all the plants .

✳ **8 APPLICATIONS :**

This webapplication can be used by farmers or users to check whether their plant is infected or not and can also show the remedy so that the user can take necessary precautions. These kind of web applications can be used in the agricultural sector as well as for small house hold plants as well.

✳ **9 CONCLUSION :**

Agriculture is the most important sector in today's life. Most plants are affected by a wide variety of bacterial and fungal diseases. Diseases on plants placed a major constraint on the production and a major threat to food security. Hence, early and accurate identification of plant diseases is essential to ensure high quantity and best quality.

In recent years, the number of diseases on plants and the degree of harm caused has increased due to the variation in pathogen varieties, changes in cultivation methods, and inadequate plant protection techniques. Usage of such applications could help the farmers to necessary precautions so that they dont face any loss as such.

✳ **10 FUTURE SCOPE :**

As of now we have just build the web application which apparently takes the input as an image and then predict the out in the near future we can develop an application which computer vision and AI techniques to predict the infection once you keep the camera near the plant or leaf this could make our project even more usable.

✳ **11 BIBILOGRAPHY :**

◆ <http://www.ijstr.org/final-print/nov2019/Fertilizers-Recommendation-System%02For-Disease-Prediction-In-Tree-Leave.pdf>

◆ <https://www.sciencedirect.com/science/article/pii/S0168169921004245>

◆ <http://www.ijetajournal.org/volume-8/issue-2/IJETA-V8I2P1.pdf>

◆ <https://www.semanticscholar.org/paper/Fertilizers-Recommendation-System%02For-Disease-In-Neela-Nithya/495379d3ef2b461fabd2de8d0605c164cb1e396f>

◆ <https://ieeexplore.ieee.org/document/8878781>

◆ <https://www.irjet.net/archives/V7/i10/IRJET-V7I1004.pdf>

☀ APPENDIX:

➤ A.Source Code:

Image Augmentation

```
In [1]: from tensorflow.keras.preprocessing.image import ImageDataGenerator

In [2]: train_datagen = ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True)
        test_datagen = ImageDataGenerator(rescale = 1./255)

In [3]: x_train = train_datagen.flow_from_directory(r'C:\Users\KUSHAL\IBM Project\Dataset Plant Disease\Veg-dataset\Veg-dataset\train_set', target_size = (128,128), batch_size=16, class_mode = 'categorical')
        x_test = test_datagen.flow_from_directory(r'C:\Users\KUSHAL\IBM Project\Dataset Plant Disease\Veg-dataset\Veg-dataset\test_set', target_size = (128,128), batch_size=16, class_mode = 'categorical')

Found 11386 images belonging to 9 classes.
Found 3416 images belonging to 9 classes.

In [4]: x_train.class_indices

Out[4]: {'Pepper__bell__Bacterial_spot': 0,
        'Pepper__bell__healthy': 1,
        'Potato__Early_blight': 2,
        'Potato__Late_blight': 3,
        'Potato__healthy': 4,
        'Tomato__Bacterial_spot': 5,
        'Tomato__Late_blight': 6,
        'Tomato__Leaf_Mold': 7,
        'Tomato__Septoria_leaf_spot': 8}

In [5]: x_test.class_indices

Out[5]: {'Pepper__bell__Bacterial_spot': 0,
        'Pepper__bell__healthy': 1,
        'Potato__Early_blight': 2,
        'Potato__Late_blight': 3,
        'Potato__healthy': 4,
        'Tomato__Bacterial_spot': 5,
        'Tomato__Late_blight': 6,
        'Tomato__Leaf_Mold': 7,
        'Tomato__Septoria_leaf_spot': 8}
```

CNN

```
In [6]: from keras.models import Sequential
        from keras.layers import Dense
        from keras.layers import Convolution2D
        from keras.layers import MaxPooling2D
        from keras.layers import Flatten

In [7]: model = Sequential()

In [8]: model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
        model.add(MaxPooling2D(pool_size=(2,2)))
        model.add(Flatten())
```

Hidden Layers

```
In [9]: model.add(Dense(300,activation='relu'))
        model.add(Dense(150,activation='relu'))
        model.add(Dense(75,activation='relu'))
```

Output Layer

```
In [10]: model.add(Dense(9,activation = 'softmax'))
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 126, 126, 32)	896
max_pooling2d (MaxPooling2D)	(None, 63, 63, 32)	0
flatten (Flatten)	(None, 127008)	0
dense (Dense)	(None, 300)	38102700
dense_1 (Dense)	(None, 150)	45150
dense_2 (Dense)	(None, 75)	11325
dense_3 (Dense)	(None, 9)	684
Total params: 38,160,755		
Trainable params: 38,160,755		
Non-trainable params: 0		

```
In [11]: model.fit_generator(x_train,steps_per_epoch=89,epochs=20,validation_data=x_test,validation_steps=27)
```

C:\Users\KUSHAL\AppData\Local\Temp\ipykernel_11676\174847055.py:1: UserWarning: 'Model.fit_generator' is deprecated and will be removed in a future version. Please use 'Model.fit', which supports generators.

```
model.fit_generator(x_train,steps_per_epoch=89,epochs=20,validation_data=x_test,validation_steps=27)
Epoch 1/20
89/89 [=====] - 32s 359ms/step - loss: 2.3261 - accuracy: 0.2784 - val_loss: 1.6143 - val_accuracy: 0.4722
Epoch 2/20
89/89 [=====] - 31s 348ms/step - loss: 1.5400 - accuracy: 0.4621 - val_loss: 1.2978 - val_accuracy: 0.6134
Epoch 3/20
89/89 [=====] - 35s 390ms/step - loss: 1.2955 - accuracy: 0.5393 - val_loss: 1.1170 - val_accuracy: 0.5787
Epoch 4/20
89/89 [=====] - 30s 341ms/step - loss: 1.1669 - accuracy: 0.5885 - val_loss: 1.0594 - val_accuracy: 0.6088
Epoch 5/20
89/89 [=====] - 35s 394ms/step - loss: 0.9271 - accuracy: 0.6601 - val_loss: 0.8662 - val_accuracy: 0.6852
Epoch 6/20
89/89 [=====] - 34s 386ms/step - loss: 0.8737 - accuracy: 0.6966 - val_loss: 0.9708 - val_accuracy: 0.6528
Epoch 7/20
89/89 [=====] - 32s 364ms/step - loss: 0.8307 - accuracy: 0.7037 - val_loss: 0.7657 - val_accuracy: 0.7014
Epoch 8/20
89/89 [=====] - 31s 344ms/step - loss: 0.6391 - accuracy: 0.7760 - val_loss: 0.6073 - val_accuracy: 0.8032
Epoch 9/20
89/89 [=====] - 31s 344ms/step - loss: 0.6904 - accuracy: 0.7633 - val_loss: 0.5604 - val_accuracy: 0.8148
Epoch 10/20
89/89 [=====] - 32s 362ms/step - loss: 0.6474 - accuracy: 0.7619 - val_loss: 0.5054 - val_accuracy: 0.8148
Epoch 11/20
89/89 [=====] - 29s 321ms/step - loss: 0.5790 - accuracy: 0.7921 - val_loss: 0.4577 - val_accuracy: 0.8218
Epoch 12/20
89/89 [=====] - 30s 338ms/step - loss: 0.6392 - accuracy: 0.7718 - val_loss: 0.7557 - val_accuracy: 0.7130
Epoch 13/20
89/89 [=====] - 29s 331ms/step - loss: 0.5791 - accuracy: 0.7990 - val_loss: 0.4456 - val_accuracy: 0.8380
Epoch 14/20
89/89 [=====] - 29s 321ms/step - loss: 0.4888 - accuracy: 0.8294 - val_loss: 0.9490 - val_accuracy: 0.6944
Epoch 15/20
89/89 [=====] - 30s 342ms/step - loss: 0.5469 - accuracy: 0.8146 - val_loss: 0.6525 - val_accuracy: 0.7708
Epoch 16/20
89/89 [=====] - 31s 343ms/step - loss: 0.5425 - accuracy: 0.8062 - val_loss: 0.5551 - val_accuracy: 0.8171
Epoch 17/20
89/89 [=====] - 33s 369ms/step - loss: 0.5371 - accuracy: 0.8118 - val_loss: 0.5010 - val_accuracy: 0.8171
Epoch 18/20
89/89 [=====] - 31s 353ms/step - loss: 0.5024 - accuracy: 0.8216 - val_loss: 0.3759 - val_accuracy: 0.8681
Epoch 19/20
89/89 [=====] - 32s 365ms/step - loss: 0.4332 - accuracy: 0.8476 - val_loss: 0.5539 - val_accuracy: 0.8009
Epoch 20/20
89/89 [=====] - 32s 365ms/step - loss: 0.4947 - accuracy: 0.8223 - val_loss: 0.5573 - val_accuracy: 0.8241
<keras.callbacks.History at 0x23c80086370>
```

Out[11]:

Saving The Model

```
In [12]: model.save("vegetable.h5")
```

Test the model

```
In [13]: import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
```

Load the model

```
In [14]: model = load_model("vegetable.h5")
```

```
In [15]: img = image.load_img("C:\\Users\\KUSHAL\\IBM Project\\Dataset Plant Disease\\Veg-dataset\\Veg-dataset\\train_set\\Pepper,_bell___healthy\\0a3f2927-4410-46a3-bfda-5f4769a5aaf8___3R_HL_8275.JPG", ta
img
```



```
In [16]: x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
pred = np.argmax(model.predict(x),axis=1)

index=['Pepper,_bell___Bacterial_spot',
'Pepper,_bell___healthy',
'Potato___Early_blight',
'Potato___Late_blight',
'Potato___healthy',
'Tomato___Bacterial_spot',
'Tomato___Late_blight',
'Tomato___Leaf_Mold',
'Tomato___Septoria_leaf_spot']

index[pred[0]]

1/1 [=====] - 1s 747ms/step
```

Out[16]: 'Pepper,_bell___healthy'

```
In [17]: img = image.load_img("C:\\Users\\KUSHAL\\IBM Project\\Dataset Plant Disease\\Veg-dataset\\Veg-dataset\\train_set\\Tomato___Septoria_leaf_spot\\00f16858-f392-4d9e-ad9f-efab8049a13f___3R_Sept.L.S.8
img
```



```
In [18]: x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
pred = np.argmax(model.predict(x),axis=1)

index=['Pepper,_bell___Bacterial_spot',
'Pepper,_bell___healthy',
'Potato___Early_blight',
'Potato___Late_blight',
'Potato___healthy',
'Tomato___Bacterial_spot',
'Tomato___Late_blight',
'Tomato___Leaf_Mold',
'Tomato___Septoria_leaf_spot']

index[pred[0]]

1/1 [=====] - 0s 138ms/step
```

Out[18]: 'Tomato___Septoria_leaf_spot'

