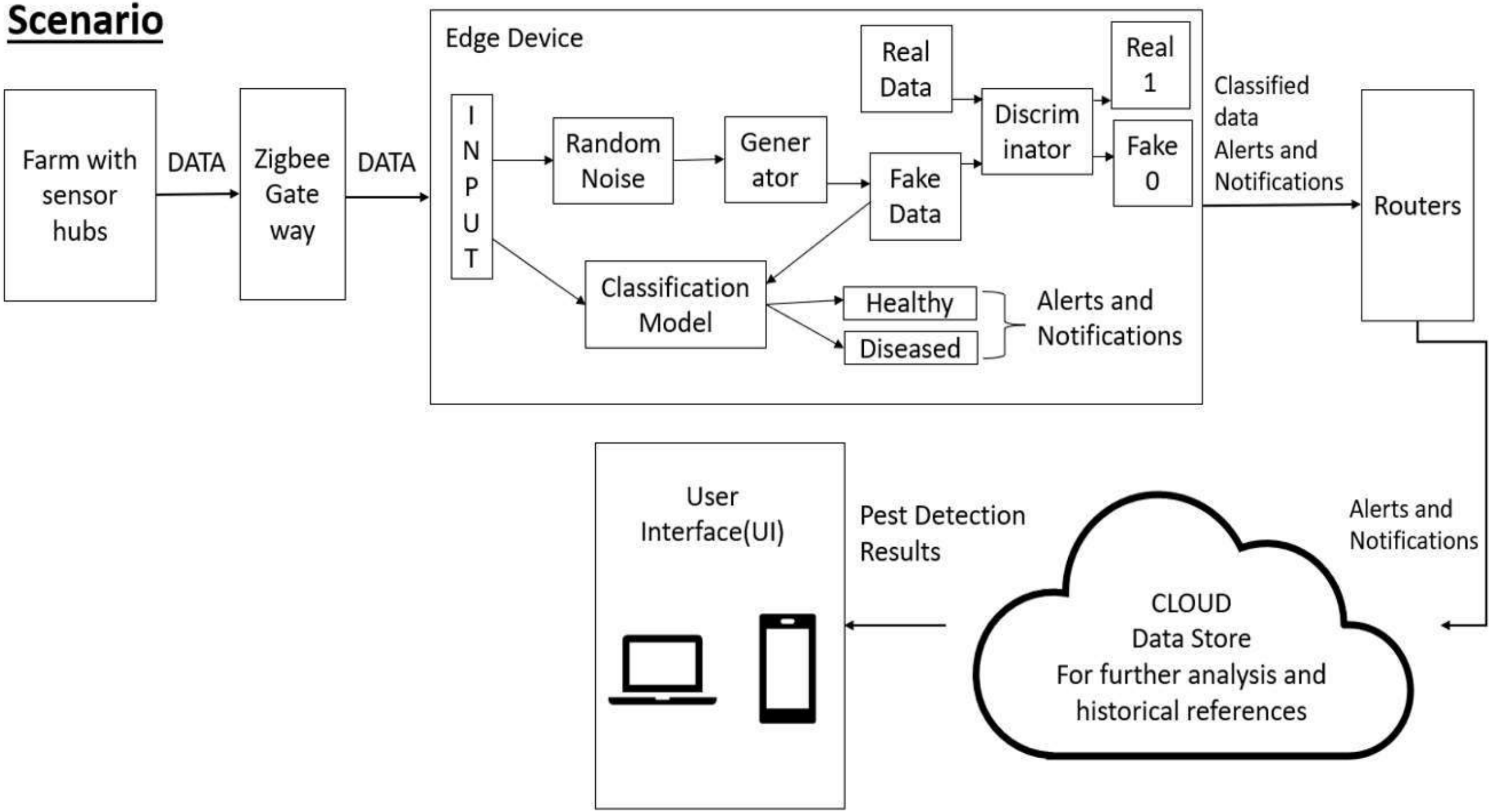
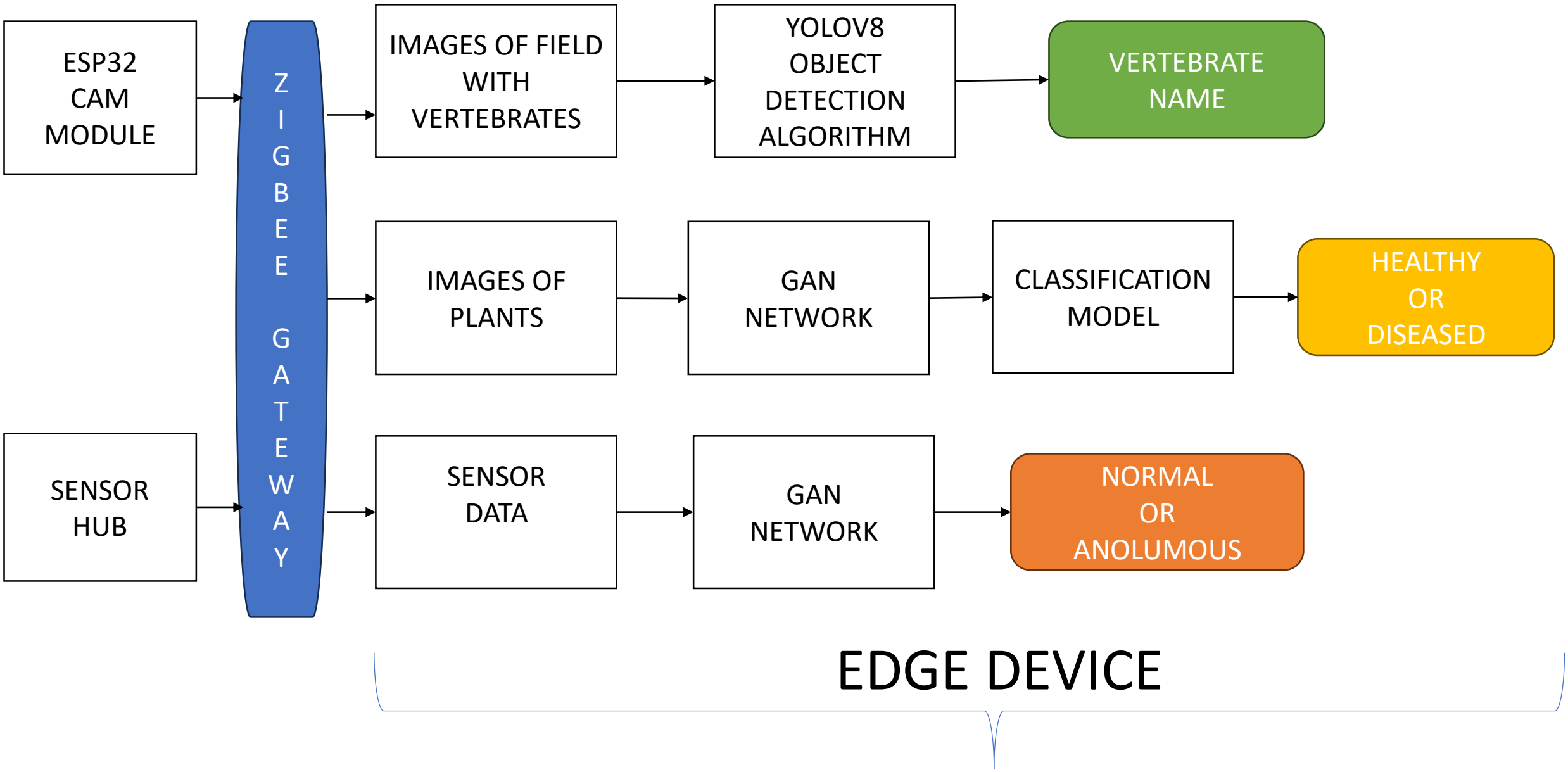
Generative Adversarial Networks (GANs) for Smart Agriculture

Generative Adversarial Networks (GANs)

- GAN offer a novel method for image augmentation through generative model learning of underlying distributions of training data.
- **SCOPE:**
  - Data Augmentation
  - Feature extraction
  - Real-time monitoring
  - Supplements limited real-world data for improved ML model performance.
  - Plants are classified as ill or healthy based on both real and synthetic data.

Scenario


Farm with sensor hubs
DATA
Zigbee Gate way
DATA
Edge Device
INPUT
Random Noise
Gener ator
Fake Data
Real Data
Discrim inator
Real 1
Fake 0
Classification Model
Healthy
Diseased
Alerts and Notifications
Classified data Alerts and Notifications
Routers
User Interface(UI)
Pest Detection Results
CLOUD
Data Store
For further analysis and historical references
Alerts and Notifications

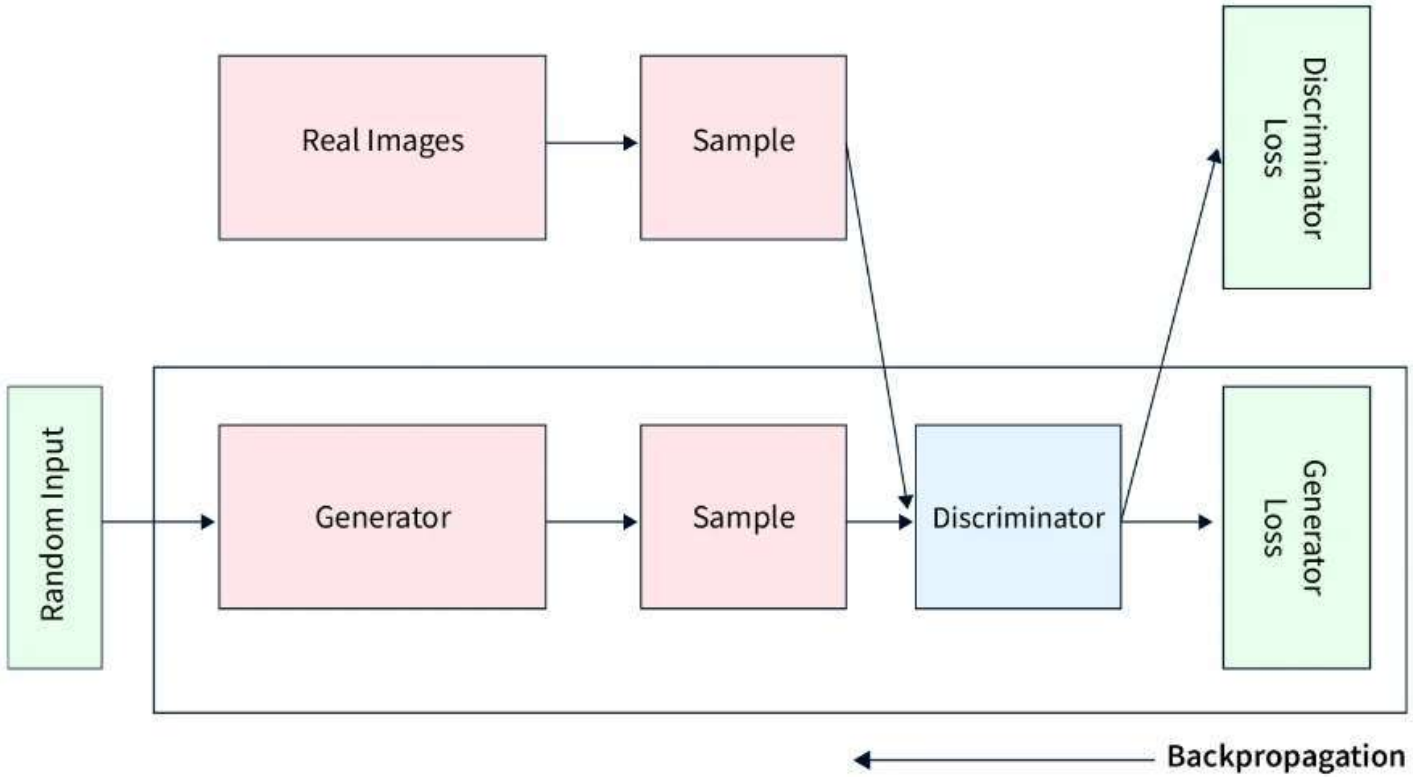

ESP32 CAM MODULE
SENSOR HUB
ZIGBEE GATEWAY
IMAGES OF FIELD WITH VERTEBRATES
IMAGES OF PLANTS
SENSOR DATA
YOLOV8 OBJECT DETECTION ALGORITHM
GAN NETWORK
GAN NETWORK
VERTEBRATE NAME
CLASSIFICATION MODEL
HEALTHY OR DISEASED
NORMAL OR ANOLUMOUS
EDGE DEVICE

Approach

- GANs are able to learn from data that is **not labeled.**
- Able to generate data that is **more diverse ,more realistic** than other models.

How are we Going to Build This?

1) Define generator and discriminator architecture
2) Compile model with Adam optimizer.
3) Generate random input for generator.
4) Generate fake images with generator.
5) Combine fake and real images.
6) Train discriminator on combined images.
7) Discriminator classify image as real or fake.
8) Generate new random noise.
9) Generate fake images with generator.
10) Train generator using discriminator predictions.
11) Repeat steps 3-9 for satisfactory results.



Numerical data

- Sensor Data:
- https://drive.google.com/drive/folders/11CoI7J7rYyRr3GDk_92sf6xrh5Z-mFFk?usp=sharing
- **CTGAN:**
  - It is a collection of Deep Learning based synthetic data generators for single table data
  - They are able to learn from real data and generate synthetic data with high fidelity.
- An efficient intrusion detection system (IDS) that assures **secure and reliable data transmission** from IoT devices (IoTDs) by detecting **anomalous activities** such as sending data which is different from their normal state.
- The initial stage of GAN can be used as IDS for the IoT devices.

REAL DATA

	A	B	C	D	E
1	Humidity	Temperature	Gas Sensor	Light Intensity	Pressure
2	70.65	28	718.55	23.85	909
3	70.04	28	718.34	22.26	909
4	70.38	28	718.12	22.98	909
5	70.24	28	718.65	23.46	909
6	70.66	28	719.98	23.57	909
7	70.75	28	719.18	22.39	909
8	70.12	28	719.92	23.06	909
9	70.09	28	719.22	22.01	909
10	70.25	28	718.76	22.18	909

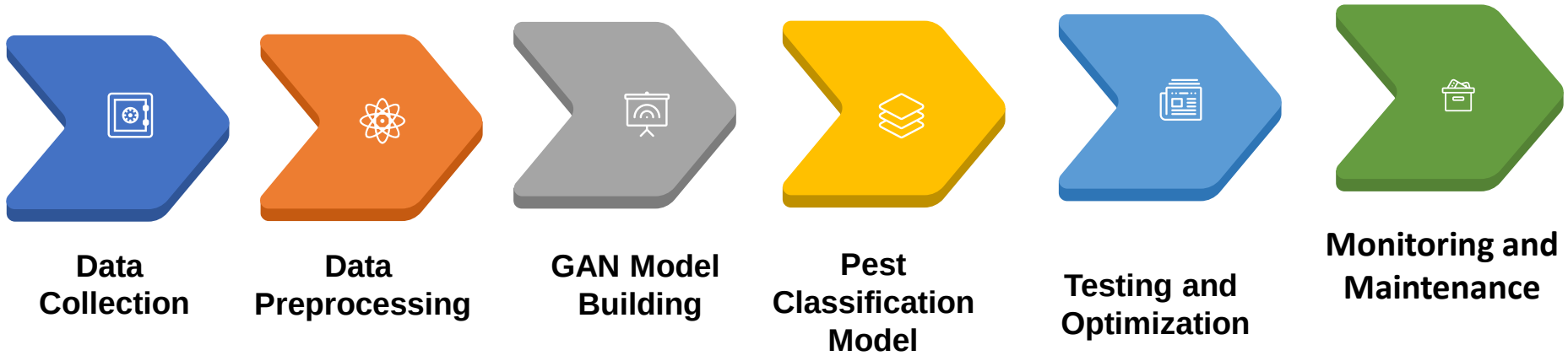
SYNTHETIC DATA

	A	B	C	D	E
1	Humidity	Temperature	Gas Sensor	Light Intensity	Pressure
2	70.03707	27.99879812	719.6142772	23.42542095	909.0021
3	70.18846	28.00028481	719.4900868	22.30806461	908.9987
4	70.47891	28.00120909	719.4344647	22.53408932	908.999
5	70.14627	28.00152682	718.3746904	22.02423578	908.9997
6	70.8778	28.00219448	718.4099276	22.98828597	909.0013
7	70.26139	28.00147668	718.6626746	22.25762252	908.9997
8	70.10859	27.99948975	719.4857003	24.12050296	908.9984
9	69.88356	28.00053439	719.7347649	23.82849943	909.0008
10	70.35546	28.00070818	718.1507561	22.80634705	909.0028

Workplan

IMAGE DATA:

- Collecting a dataset (Kaggle)
- Data Preprocessing
- GAN Model Building
- Pest Classification Model
- Testing and Optimization
  - ➢ Hyper parameter tuning
- Monitoring and Maintenance

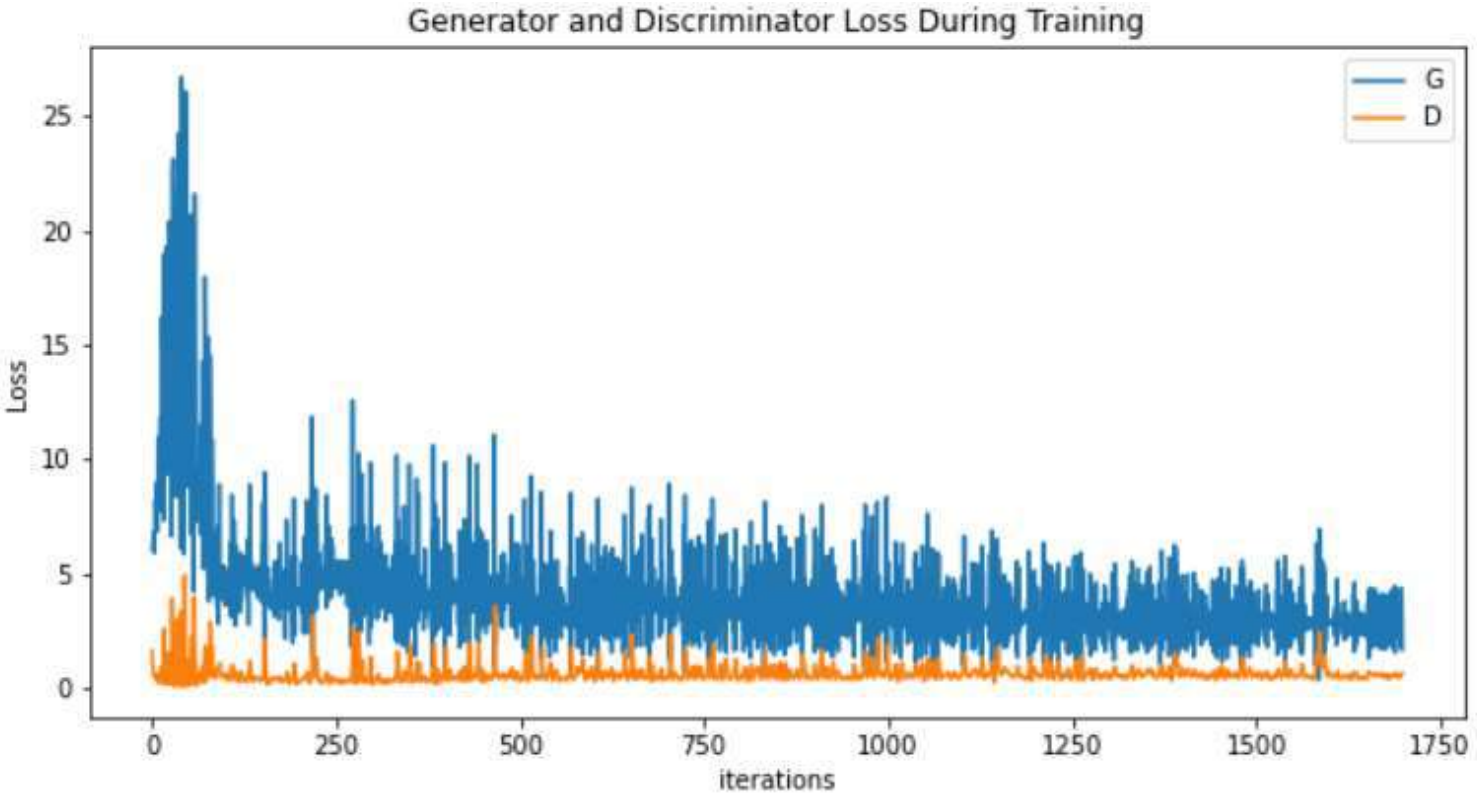


DCGAN

- **Deep convolution generative adversial networks**
- The discriminator is made up of strided convolution layers, batch norm layers, and LeakyReLU activations.
- The input is a 3x64x64 input image and the output is a scalar probability that the input is from the real data distribution.
- The generator is comprised of convolutional-transpose layers, batch norm layers, and ReLU activations.
- The input is a latent vector, that is drawn from a standard normal distribution and the output is a 3x64x64 RGB image.
- Image Data:
- https://www.kaggle.com/datasets/arjuntejaswi/plant-village
- GAN Model:
- https://drive.google.com/drive/folders/1KBey-vO_pKv7h-eGpt6BA3mlRXAdNVxh?usp=sharing

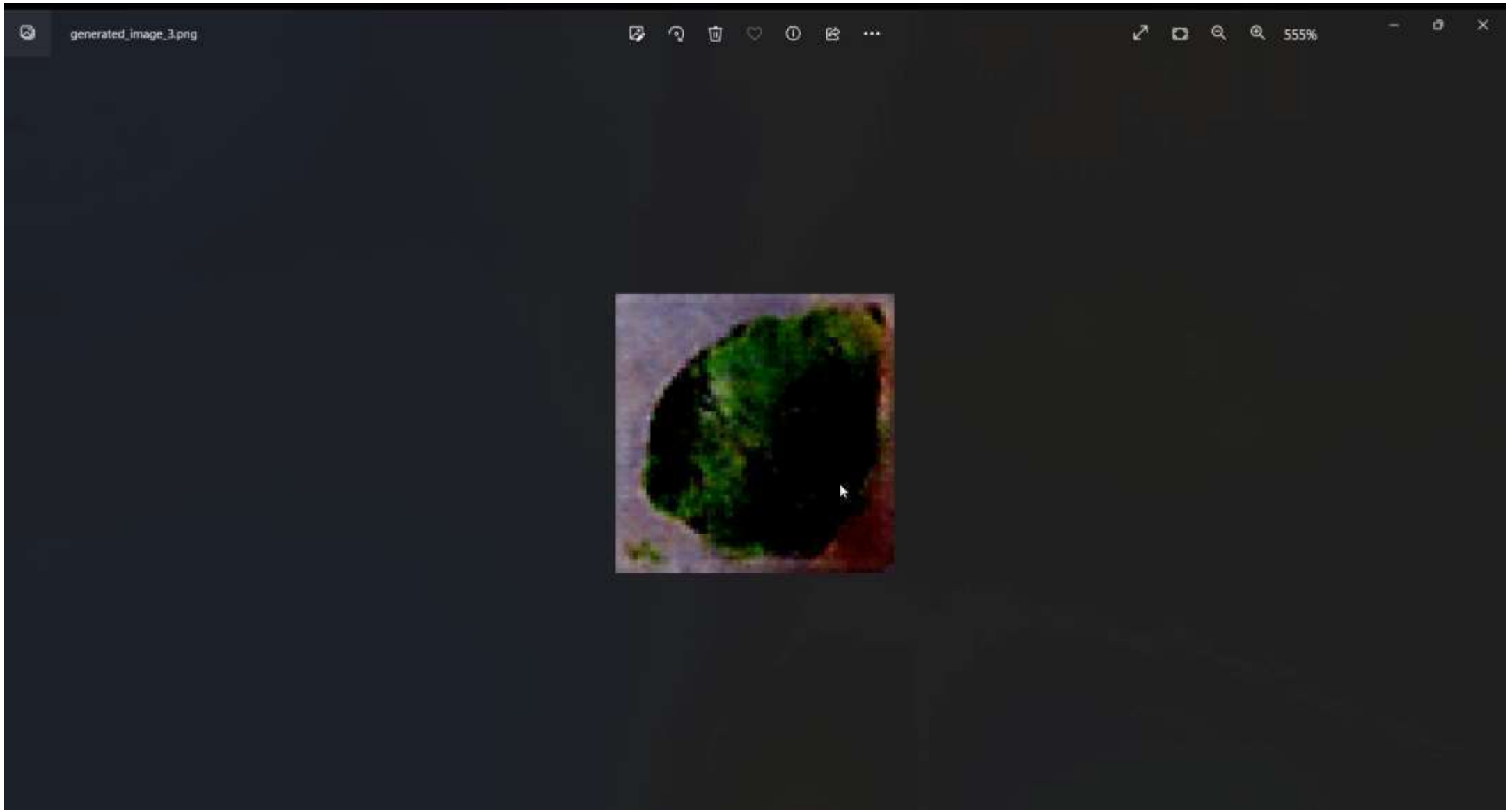
LOSS FUNCTION

$$\min_{G}\max_{D} V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}\left[logD(x)\right] + \mathbb{E}_{z\sim p_z(z)}\left[log(1-D(G(z)))\right]$$



Observation: The generator loss is not converging, while the discriminator loss is converging. As we understand, the generator and the discriminator operate adversarial, meaning they play opposing roles in a generative adversarial network (GAN).

Results



GAN Training Challenges

Now that we have finished training DCGAN with color images. Let's discuss some of the common challenges of GAN training.

GANs are very difficult to train, and here are some of the well-known challenges:

1.Non-convergence: instability, vanishing gradients, or slow training

2.Mode collapse

3.Difficult to evaluate

Failure to Converge

- Unlike training other models such as an image classifier, the losses or accuracy of D and G during training only measure D and G individually and **doesn't measure the GAN overall performance** and how good the generator is at creating images.

GAN training instability:

- Looking at the losses during training, you will notice they may oscillate wildly. And both D and G could get stuck and never improve. **Training for a long time doesn't always make the generator better.**
- The **image quality by the generator may deteriorate over time**.

Difficult to Evaluate

- It's challenging to evaluate the GAN models because there is no easy way to determine whether a generated image is "good."
- Unlike an image classifier, the prediction is either correct or incorrect according to the ground truth label.
- This leads to the discussion below on how we evaluate GAN models.

GAN Evaluation Metrics

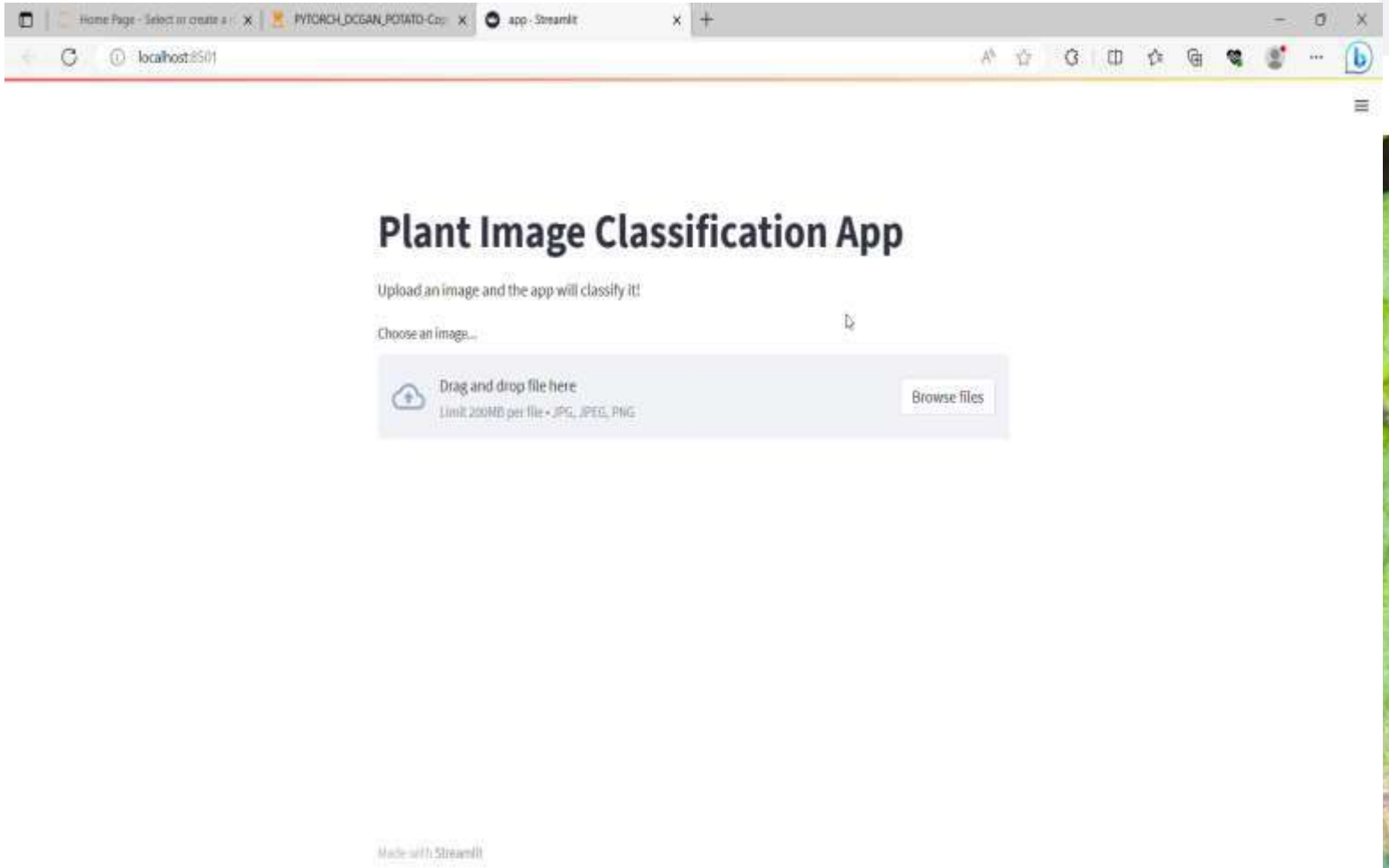
- There are two criteria for a successful generator — it should generate images with:
  1. good quality: high fidelity and realistic,
  2. diversity (or variety): a good representation of the training images' different types (or categories).

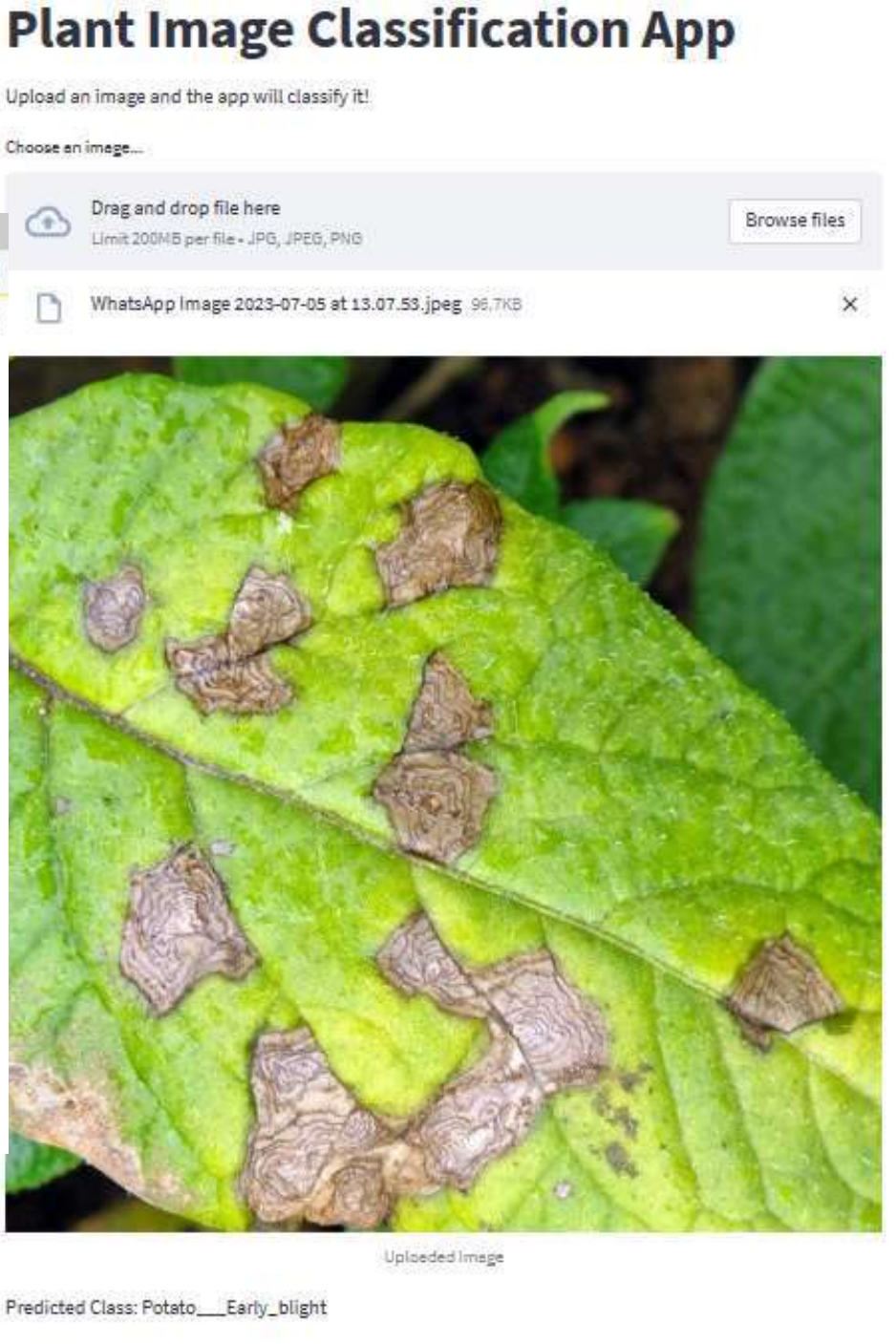
Classification model

- **Pest Classification:**
  - Utilize the GAN model to classify the sensor data received from the agricultural environment.
  - Based on the generated output, classify the data instances as either pest presence or absence.
- **K-means Algorithm:**
  - Unsupervised learning technique used for partitioning data into 'k' clusters
  - Each cluster represents a distinct group with similar characteristics
  - Extracting relevant features from the data
  - Clustering the images into two categories: healthy and diseased

- **Classification Model:** https://drive.google.com/drive/folders/1bIQSCDFi3qT-1an60ezgPtzGIZ6R5N9C?usp=sharing

Deploying the Classification model using Stream-lit

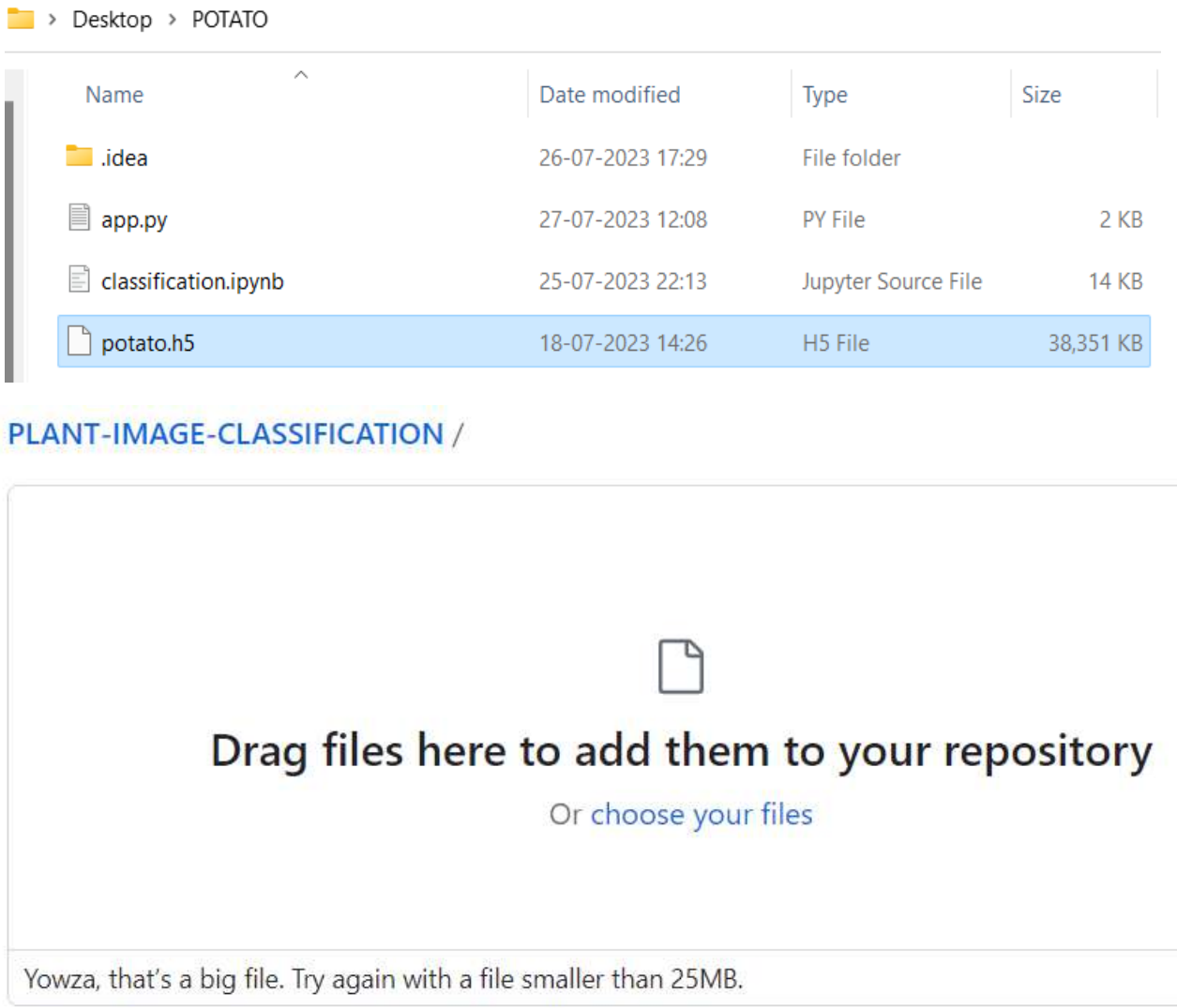




Creating Public URL With Stream-lit cloud

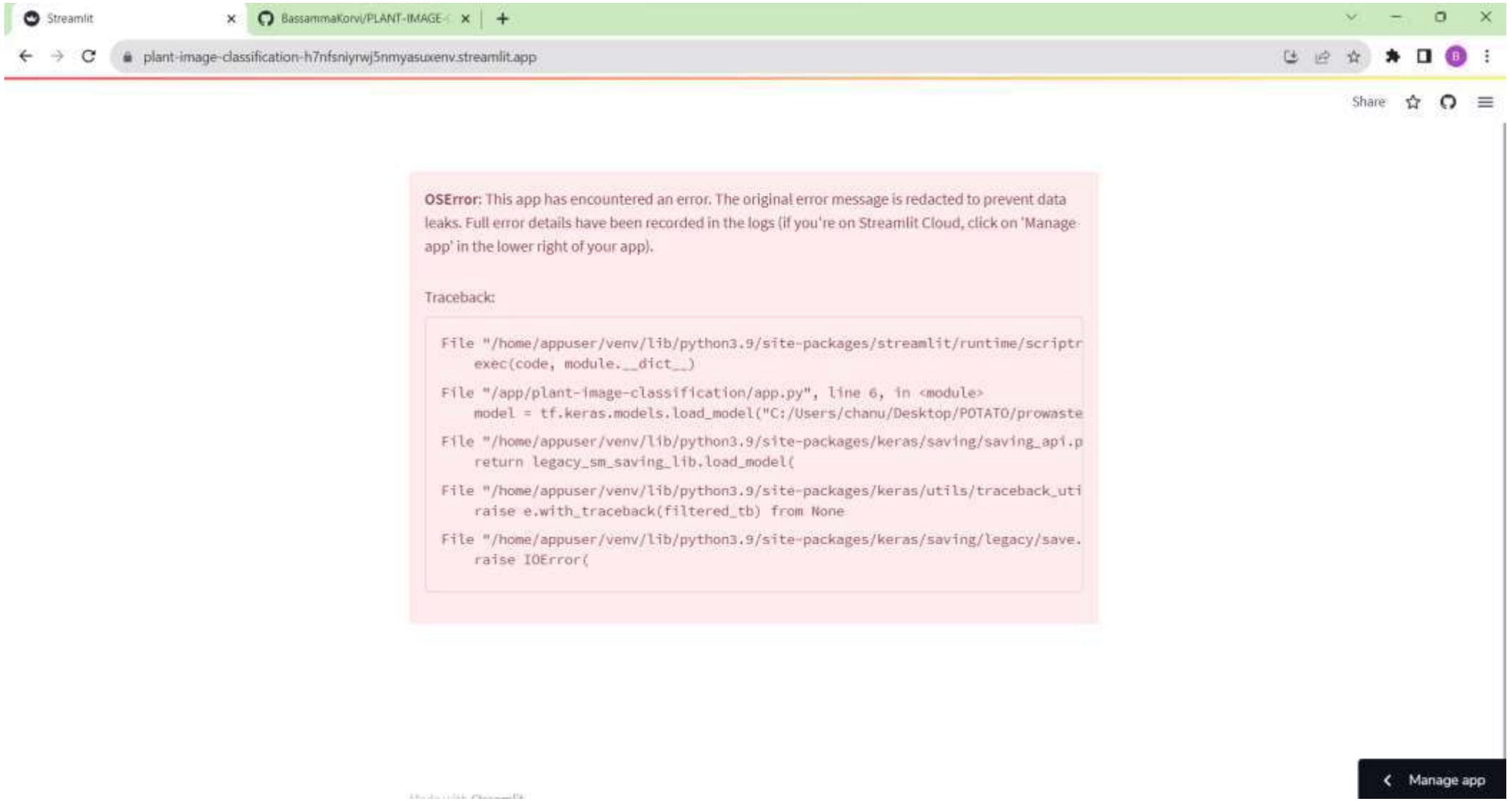
- As we developed the classification model and saved it in .h5 file format, we encountered an issue when trying to push it to a Git repository.
- The file size exceeded the limit of 25 MB set by Git, preventing us from uploading it.
- Consequently, we need to explore alternative methods for deploying the application and making it accessible through a public URL.

https://github.com/BassammaKorvi/PLANT-IMAGE-CLASSIFICATION/tree/main



Creating Public URL With Stream-lit cloud

https://plant-image-classification-h7nfsniyrwj5nmyasuxenv.streamlit.app/



Learning Outcomes

- Embrace learning and stay open to acquiring new skills.
- Recognize the importance of effective communication for success.
- Welcome and value feedback for personal growth and improvement.
- Explore a diverse range of opportunities to gain experience and clarify goals.
- Put knowledge and skills into practice through practical engagement.
- Adapt to new teams and working environments with flexibility and resourcefulness.
- resourcefulness, creativity, and problem-solving.

References

- https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html
- **Synthesis of IoT Sensor Telemetry Data for Smart Home Edge-IDS Evaluation**
  Sasirekha GVK, Amulya Bangari, Madhav Rao, Jyotsna Bapat, Debabrata Das
- **SMOTE: Synthetic Minority Over-sampling Technique** N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer
- **Copula Flows for Synthetic Data Generation**
- Sanket Kamthe, Samuel Assefa, Marc Deisenroth
- **Are GANs Created Equal? A Large-Scale Study**
  Mario Lucic , Karol Kurach , Marcin Michalski ,Olivier Bousquet , Sylvain Gelly
- Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks.

THANK YOU