# Machine Learning Project Documentation

## Model Refinement

### Overview

The model refinement phase is crucial in improving the performance of the machine learning model. It involves iterative processes of evaluating and adjusting the model to enhance its accuracy and robustness.

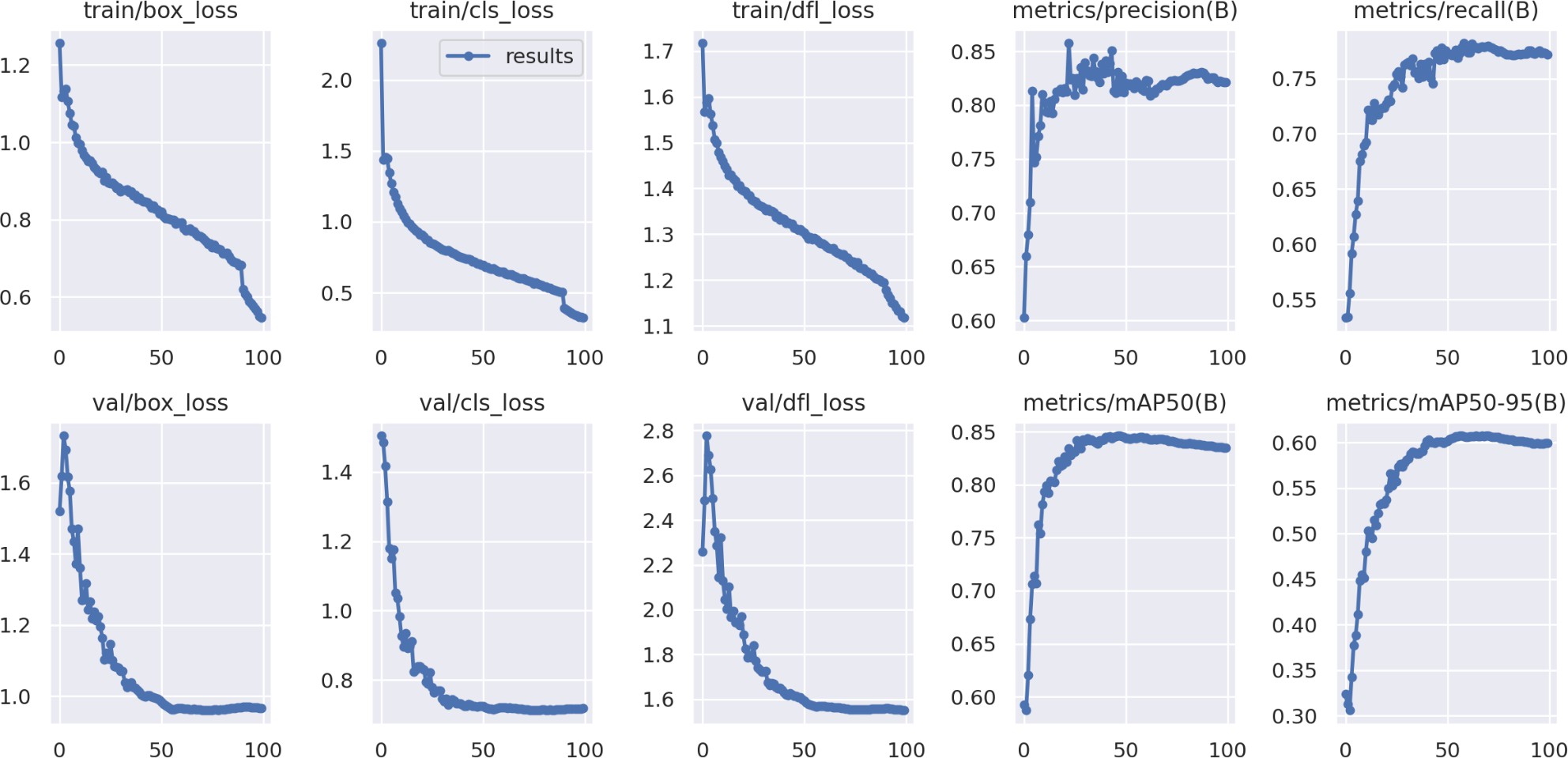
### Model Evaluation

Initial model evaluation indicated areas needing improvement. Key metrics and visualizations, such as confusion matrices and loss curves, highlighted issues in classifying certain diseases and suggested overfitting.

**Confusion Matrix:**A chart with blue squares

Description automatically generated

**Loss and Metric Curves:**



### Refinement Techniques

Several techniques were employed to refine the model, including:

* Hyperparameter Tuning: Adjusting learning rates, batch sizes, and epochs.
* Algorithm Testing: Exploring different architectures within the YOLO v8 framework.
* Ensemble Methods: Combining multiple models to leverage their strengths.

## Hyperparameter Tuning

Hyperparameter tuning involved experimenting with various configurations to find the optimal settings. Key adjustments included:

* Learning Rate: Lowering the learning rate to reduce overfitting.
* Batch Size: Increasing batch size to stabilize training.
* Epochs: Extending training epochs for better convergence.

### Cross-Validation

Cross-validation strategies were revised to ensure robust model evaluation. Techniques such as K-fold cross-validation were employed to better generalize the model performance across different data subsets.

### Feature Selection

Feature selection methods, including recursive feature elimination and correlation analysis, were applied. These methods helped in identifying the most relevant features, thus improving model performance by reducing noise and overfitting.

## Test Submission

### Overview

The test submission phase involved preparing the model for deployment and evaluating its performance on a test dataset.

### Data Preparation for Testing

The test dataset was prepared by applying the same preprocessing steps as the training data, including resizing, normalization, and data augmentation.

### Model Application

The trained model was applied to the test dataset to generate predictions. Below is a code snippet demonstrating this process with YOLO v8:

A screenshot of a computer program

Description automatically generatedA computer screen shot of a black screen

Description automatically generated

## Test Metrics

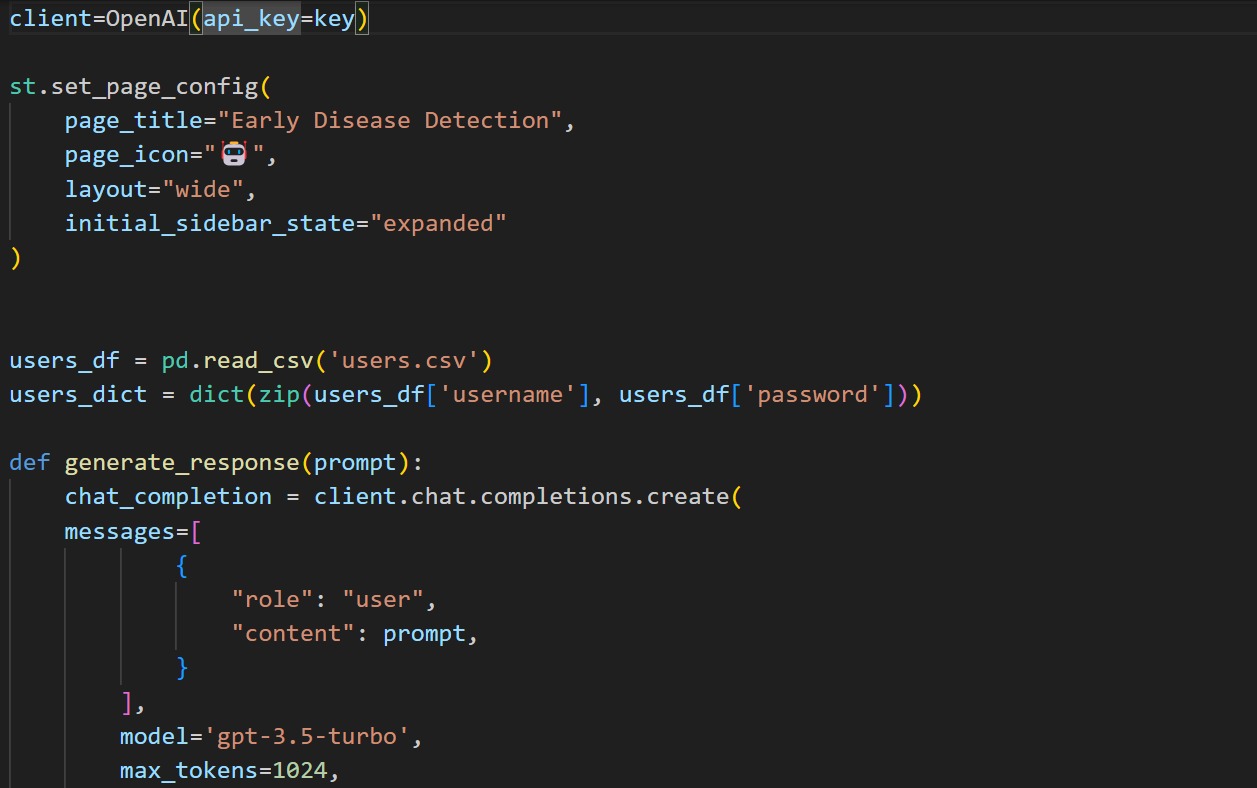
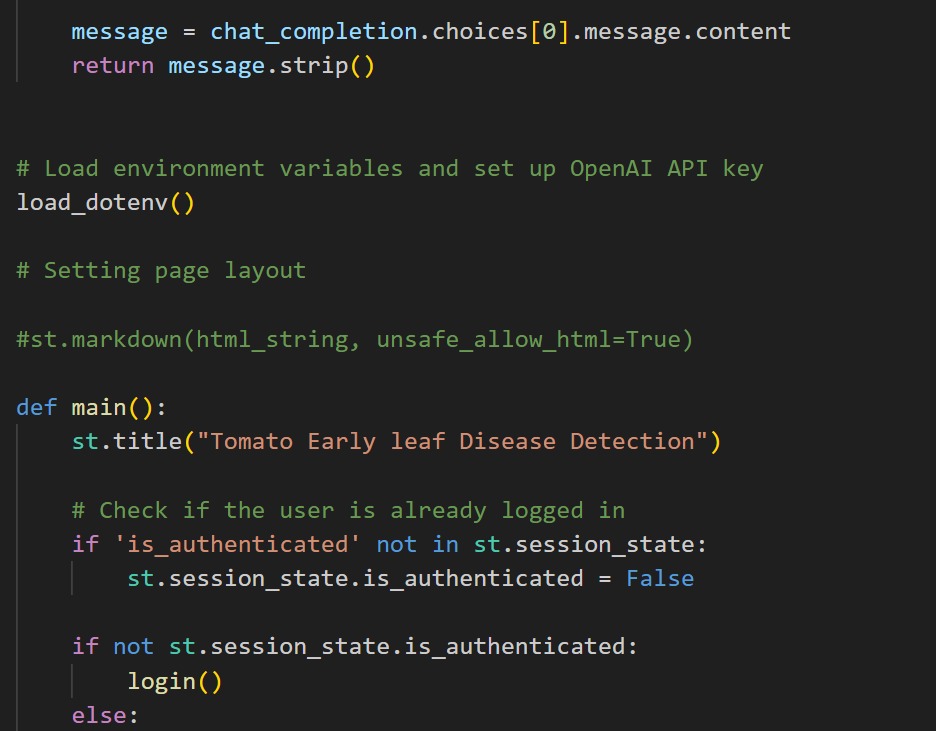
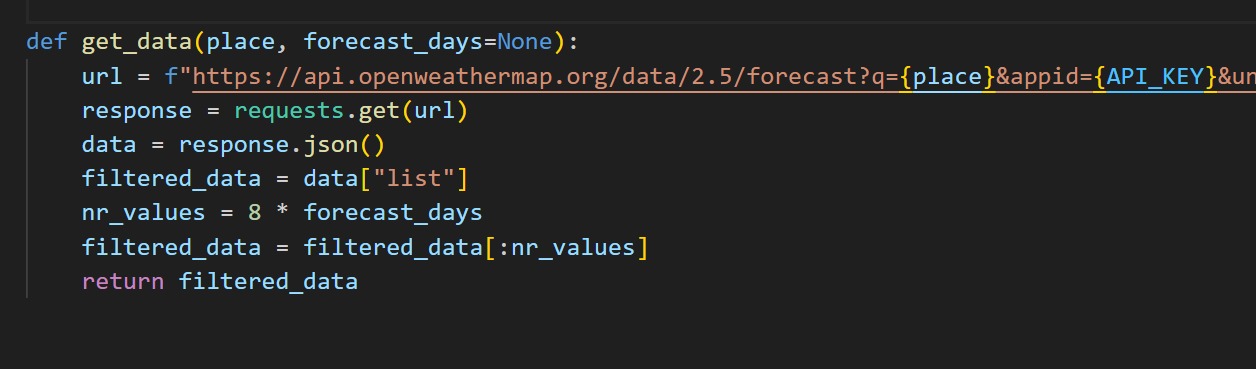
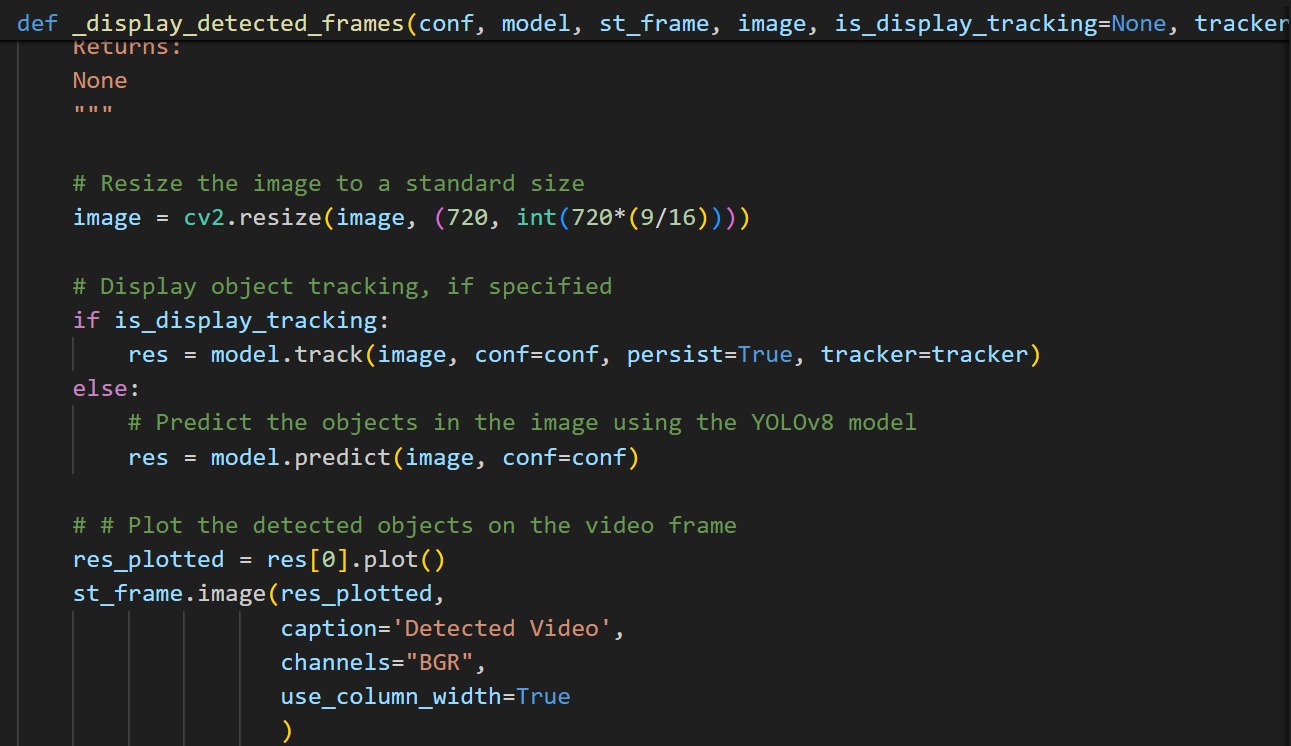
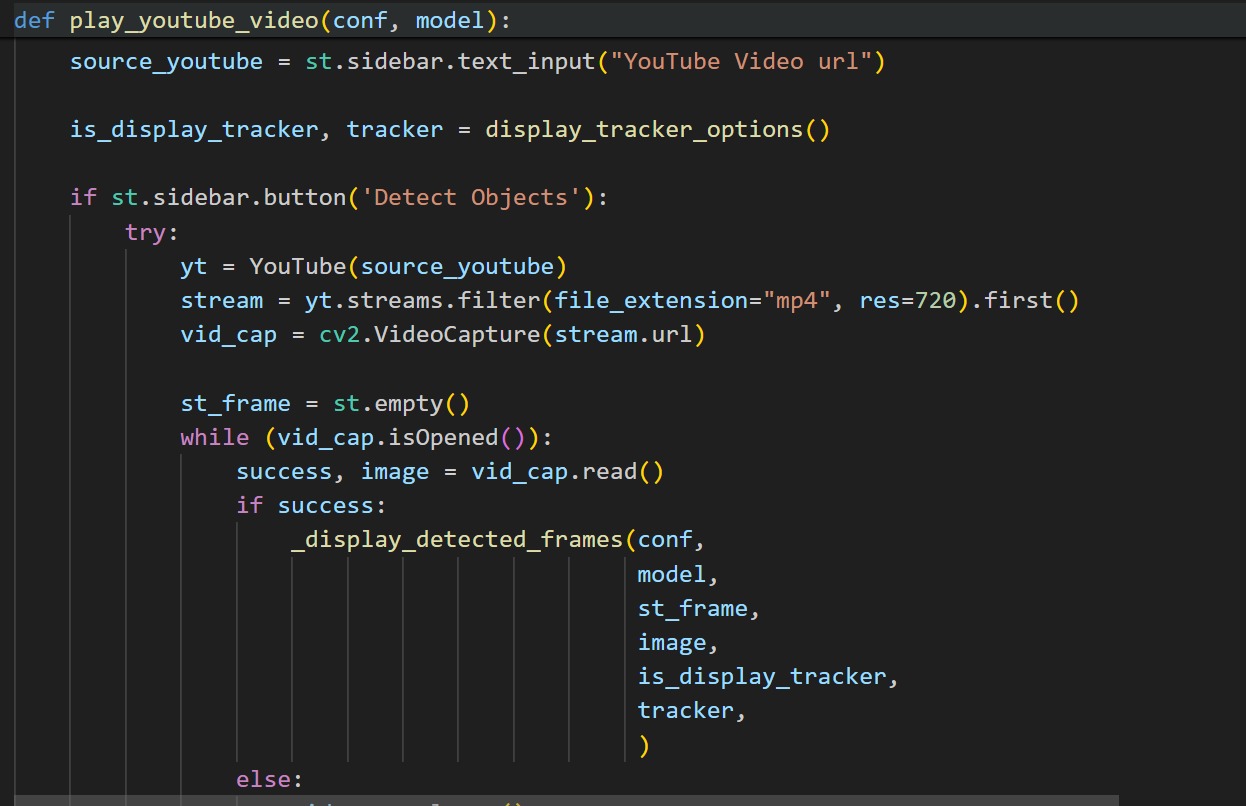
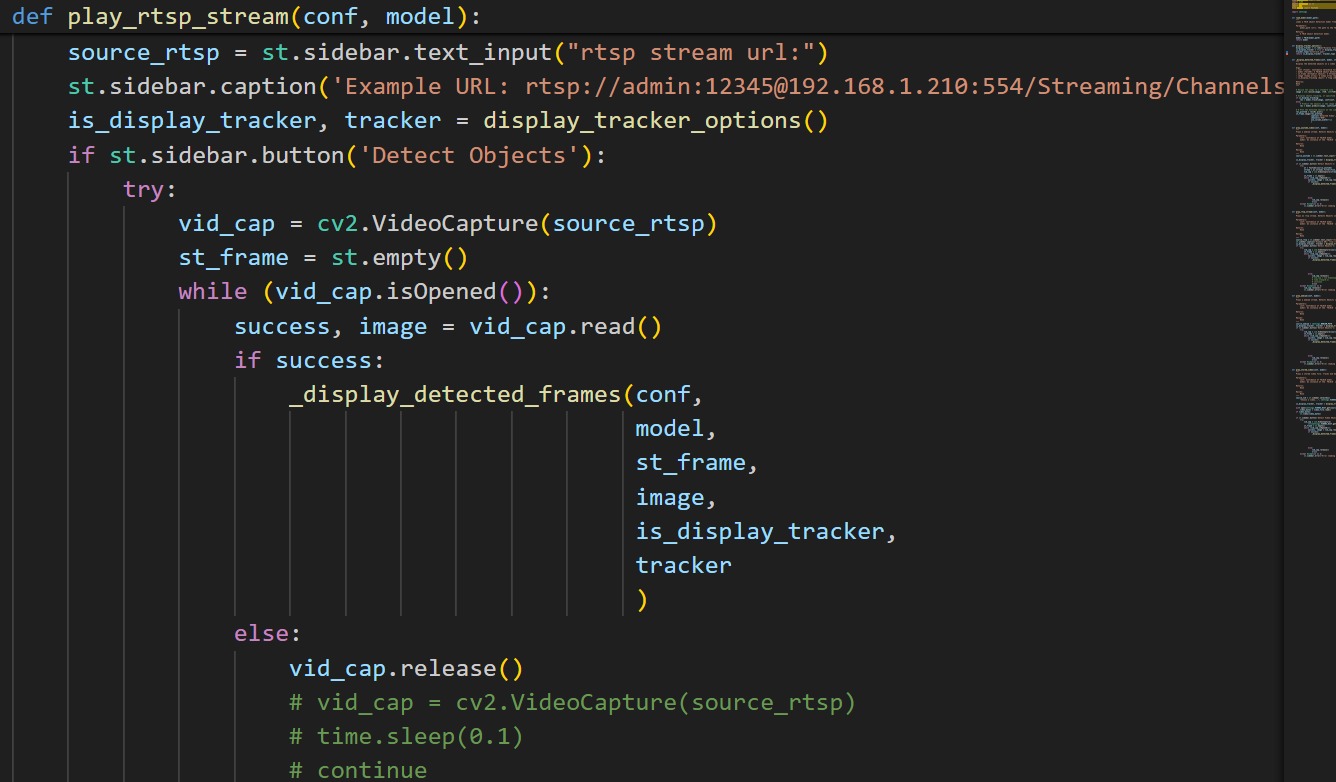
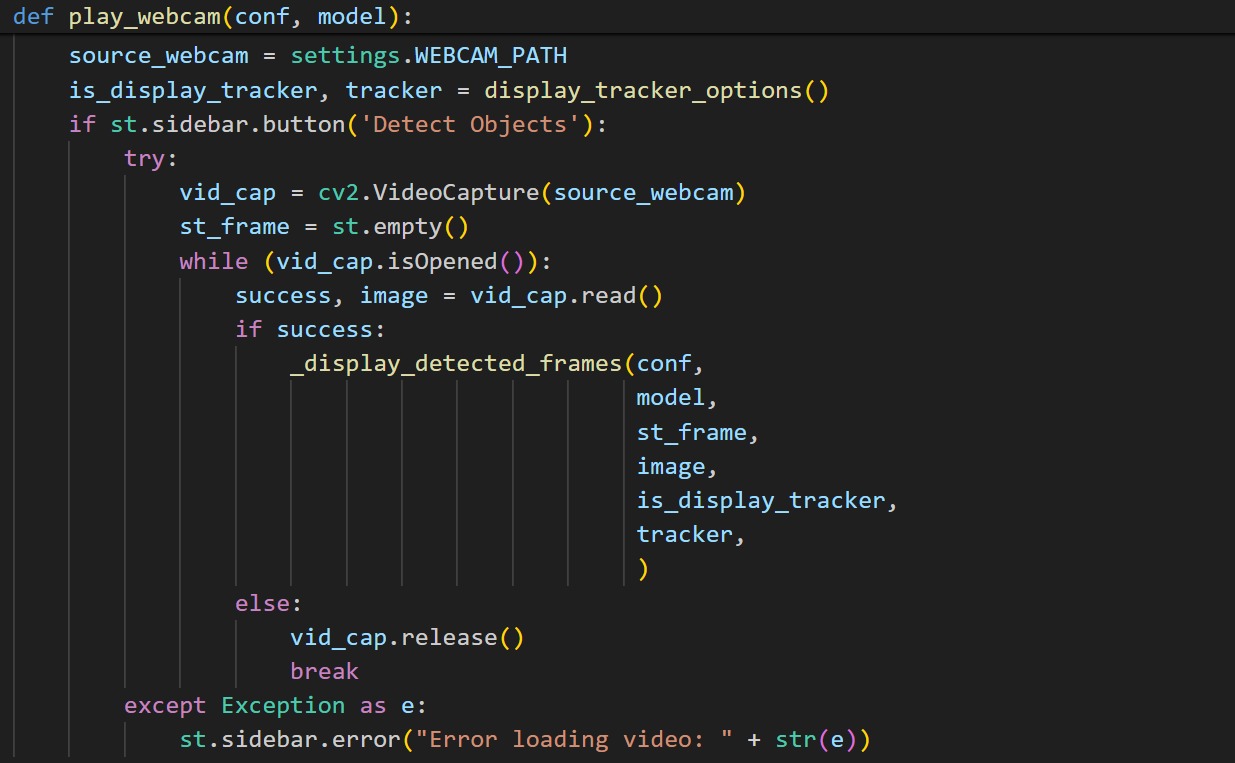
The performance metrics on the test dataset included precision, recall, and mAP (mean Average Precision). These results were compared with training and validation metrics to ensure consistency.

## Model Deployment

The model was deployed using Streamlit to create an interactive user interface. This interface allowed users to upload images and receive disease diagnosis and fertilizer recommendations in real-time.

Below is a code snippet demonstrating the Streamlit deployment:

**Code Snippets:**



## Conclusion

The model refinement and test submission phases were integral to achieving a high-performance model for tomato disease detection. Despite challenges such as overfitting and misclassification, iterative adjustments and robust evaluation techniques resulted in a model capable of accurate and reliable predictions.