Background Information
For the Convolutional Neural Network (CNN) case study, the team tested three CNN models - 1, 2a and 2b to classify the fruits dataset.

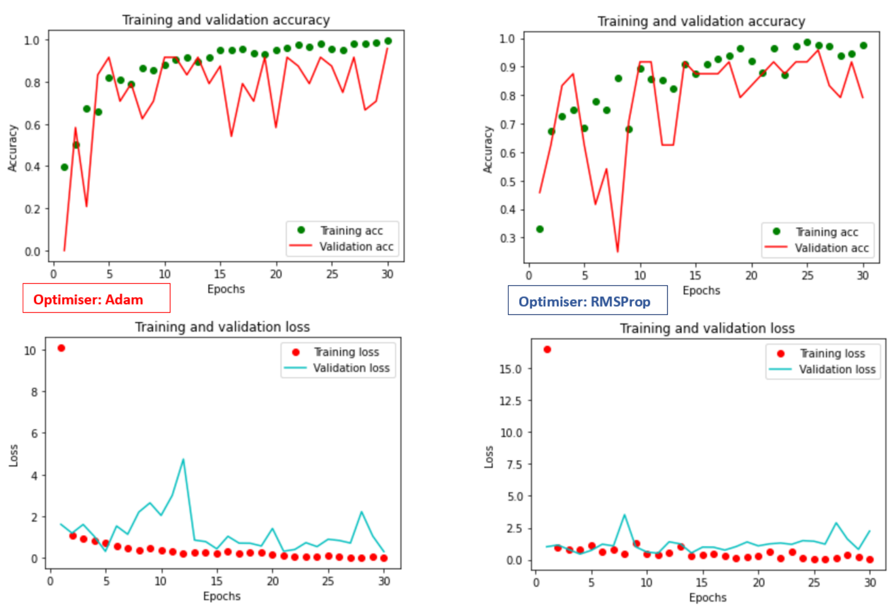
Standard Parameters

Optimization Algorithms	Adam and RMSprop
Steps	10
Epochs	30. It was observed that the model achieved consistent 100% accuracy after 25 epochs, hence we decided to cap it to 30 epochs to reduce training time.
Conv2D layers	2 layers with 64 filters and 32 layers respectively and 3x3 kernel size
Dense layers	2 layers of 128 nodes and 4 output nodes
Drop out	0.5 is selected to prevent overfitting. During training, 50% of the nodes will be randomly skipped during model training.

Model 1: **Model Training (Handpicked Validation Data)**	Implemented using the flow_from_directory method from Keras. The data is prepared and split into 4 folders which are automatically labelled when imported. We implemented the following code to verify actual results with the samples from the test folder.
Optimizer: Adam	For this model, validation data is handpicked whereas the validation data used in model 2 is randomly selected. In the train dataset, we split it into two with one set being picked as a validation dataset. Result: Accuracy of 85.5% with the train time of ~624s, at ~21s/epoch and ~2s/step We trained the model again, now using data from the test set for our validation model.Result: Accuracy of 100%, with final epoch loss: 0.0018 - accuracy: 1.0000 - val_loss: 6.7386e-05, with train time of ~685s, with 23s/epoch and ~2s/step However, actual results translated only a result of 69.1% accuracy when we evaluated the model with the test dataset.
Optimizer: RMSprop	We decided to try the RMSprop optimization algorithm, which we found to yield a better accuracy overall. Repeating the steps earlier with the Adam optimizer, Result: Accuracy of 96.43% is achieved, train time of ~647s, ~23s/epoch and ~2s/step We retrained the model, using data from the test set for our validation model. Result: Accuracy of 100%with Train time of ~720s, ~24s/epoch and ~2s/step. Loss: 0.0056 - accuracy: 1.0000 - val_loss: 1.6429e-04 - val_accuracy: 1.000 Actual results translated a result of 92.7% accuracy when we evaluated the model with the test dataset. The result yielded is significantly higher than when we used the Adam optimizer.
Model 2a: **Model Training (Randomised validation data using validation_split=0.1)**	Images loaded via self-defined loop using Pillow. For this model, we used a lower resolution of 30x30 instead of 200x200.
Optimizer: Adam	Result: Accuracy of 99.07% with Train time of ~32s, ~1s/epoch and ~42ms/step. Loss: 0.0221 - accuracy: 0.9907 - val_loss: 0.6493 - val_accuracy: 0.8750

	Actual results translated a result of 94.5% accuracy
Optimizer: RMSprop	Result: Accuracy of 98.15% with Train time of ~32s, ~1s/epoch and ~45ms/step. Loss: 0.0515 - accuracy: 0.9815 - val_loss: 0.7634 - val_accuracy: 0.9167 Actual results translated a result of 87.3% accuracy
Model 2b: **Model Training (Randomised validation data using validation_split=0.1)**	Images loaded via self-defined loop using Pillow. For this model, we used a resolution of 200x200.
Optimizer: Adam	Result: Accuracy of 98.15% with Train time of ~1170s, ~38s/epoch and ~6ms/step. Loss: 0.0575 - accuracy: 0.9815 - val_loss: 0.7634 - val_accuracy: 0.9167 Actual results translated a result of 87.3% accuracy
Optimizer: RMSprop	Result: Accuracy of 99.07% with Train time of ~1172s, ~39s/epoch and ~6s/step. Loss: 0.0585 - accuracy: 0.9907 - val_loss: 1.5175 - val_accuracy: 0.833 Actual results translated a result of 94.5% accuracy

Accuracy and Loss
Generally, all variations followed a similar training and validation accuracy and loss graph as shown below. Once a consistently high accuracy and validation accuracy is achieved, training of a model can be interrupted as made evident in our pdf file - model 1 training output. We deemed our configuration of 30 epochs, batch size of 10, and steps of 10 be capable of achieving such a result.



Training and Validation graph for model 2b

Conclusion
Comparing the results, we conclude that the RMSprop optimization algorithm yields better results on higher resolution samples as compared to the Adam optimization algorithm. We also observed that the datasets for training and testing are too small. This results in overfitting that is rectified by a drop out of 0.5. To adjust for the discrepancies due to the small dataset, we divided by the total number of test samples (n = 55) when implementing the code for the model.