Premières modélisations avec des architectures Baseline

- Architecture LeNet
- CNN
- Dense

Données utilisées

Configurations

- Images en Niveau de Gris
- Dimensions 28x28x1

Classifications étudiées

- Binaire Sain/Malade
- 4 classes : Normal / Viral_Pneumonia / Lung_Opacity / COVID

Cas 1 - Binaire : Sain / Malade

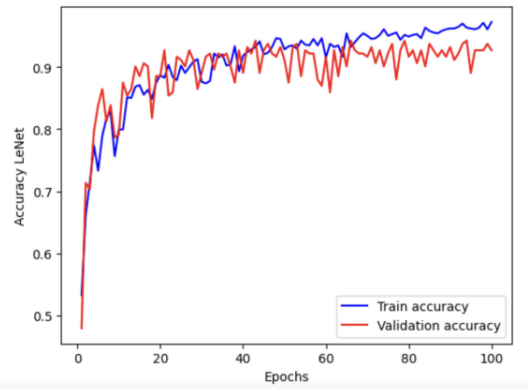
1. Taille = 28
2. Images = 600 images par classe

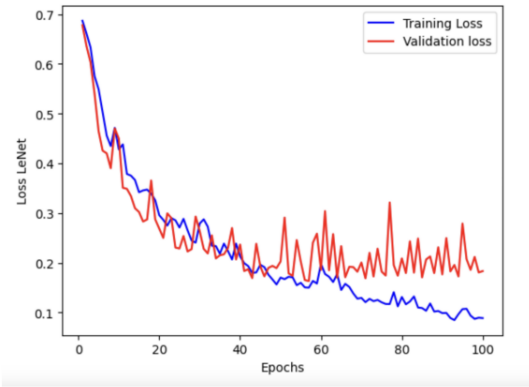
- Sain = Normal
- Malade = 1/3 COVID, 1/3 Viral Pneumonia / 1/3 Lung Opacity

Accuracy LeNet / Loss LeNet

In [37]:
```
# Charger les images
image1 = plt.imread(binaire_28_no_sparse_lenet_accuracy)
image2 = plt.imread(binaire_28_no_sparse_lenet_loss)

# Créer une figure et un ensemble d'axes
fig, axes = plt.subplots(1, 2, figsize=(16, 8))  # 1 ligne, 2 colonnes

# Afficher la première image
axes[0].imshow(image1)
axes[0].axis('off')  # Masquer les axes pour la première image

# Afficher la seconde image
axes[1].imshow(image2)
axes[1].axis('off')  # Masquer les axes pour la seconde image
```

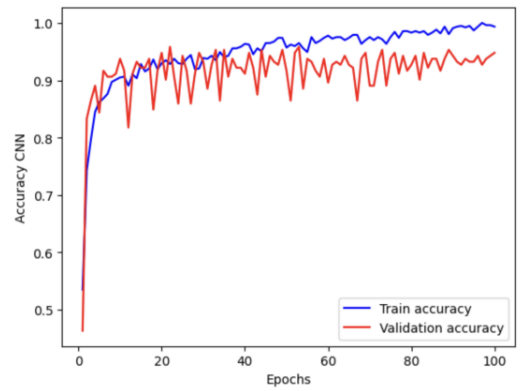
```
plt.show()
```

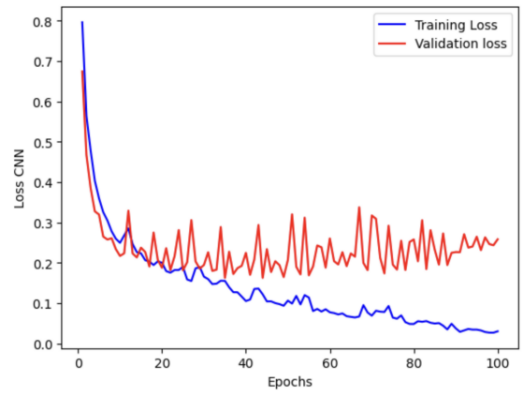




Accuracy CNN / Loss CNN

In [38]:
```
# Charger les images
image1 = plt.imread(binaire_28_no_sparse_cnn_accuracy)
image2 = plt.imread(binaire_28_no_sparse_cnn_loss)

# Créer une figure et un ensemble d'axes
fig, axes = plt.subplots(1, 2, figsize=(16, 8))  # 1 ligne, 2 colonnes

# Afficher la première image
axes[0].imshow(image1)
axes[0].axis('off')  # Masquer les axes pour la première image

# Afficher la seconde image
axes[1].imshow(image2)
axes[1].axis('off')  # Masquer les axes pour la seconde image

plt.show()
```

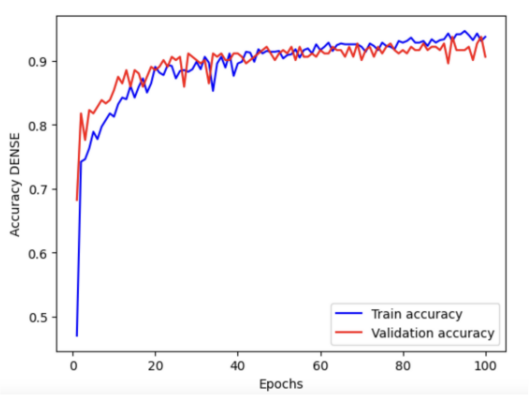


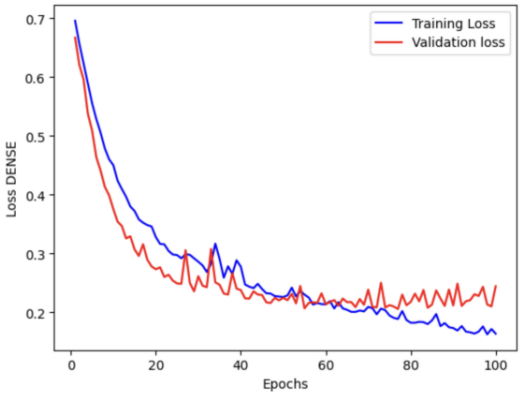


Accuracy DENSE / Loss DENSE

In [39]:
```
# Charger les images
image1 = plt.imread(binaire_28_no_sparse_dense_accuracy)
image2 = plt.imread(binaire_28_no_sparse_dense_loss)

# Créer une figure et un ensemble d'axes
fig, axes = plt.subplots(1, 2, figsize=(16, 8))  # 1 ligne, 2 colonnes

# Afficher la première image
```

```
axes[0].imshow(image1)
axes[0].axis('off')  # Masquer les axes pour la première image

# Afficher la seconde image
axes[1].imshow(image2)
axes[1].axis('off')  # Masquer les axes pour la seconde image

plt.show()
```





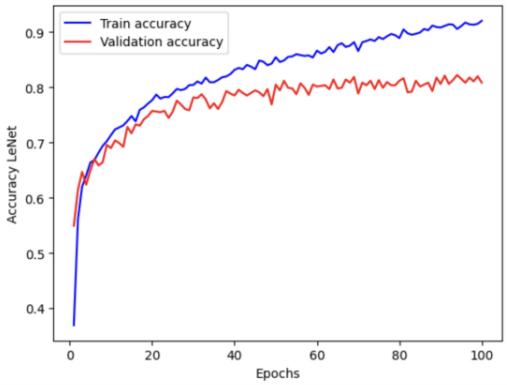
Cas 2 - Multiclasse :

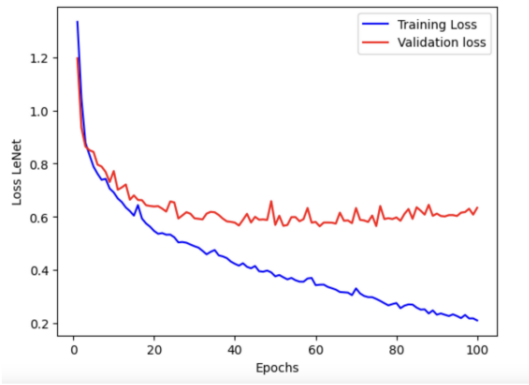
1. Taille = 28
2. Images = 1345 images par classe

- Normal
- COVID
- Viral Pneumonia
- Lung Opacity

Accuracy LeNet / Loss LeNet

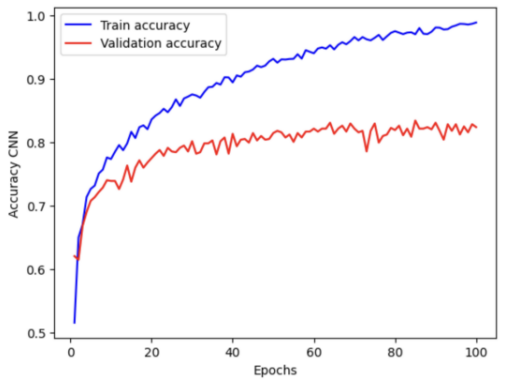
In [43]:
```
# Charger les images
image1 = plt.imread(mc_28_sparse_lenet_accuracy)
image2 = plt.imread(mc_28_sparse_lenet_loss)

# Créer une figure et un ensemble d'axes
fig, axes = plt.subplots(1, 2, figsize=(16, 8))  # 1 ligne, 2 colonnes

# Afficher la première image
axes[0].imshow(image1)
axes[0].axis('off')  # Masquer les axes pour la première image

# Afficher la seconde image
axes[1].imshow(image2)
axes[1].axis('off')  # Masquer les axes pour la seconde image

plt.show()
```

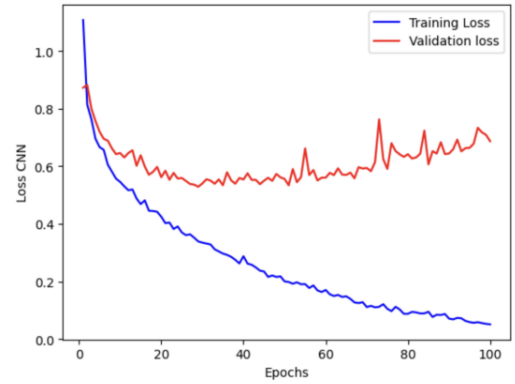




Accuracy CNN / Loss CNN

In [44]:
```
# Charger les images
image1 = plt.imread(mc_28_sparse_cnn_accuracy)
image2 = plt.imread(mc_28_sparse_cnn_loss)

# Créer une figure et un ensemble d'axes
fig, axes = plt.subplots(1, 2, figsize=(16, 8))  # 1 ligne, 2 colonnes

# Afficher la première image
axes[0].imshow(image1)
axes[0].axis('off')  # Masquer les axes pour la première image

# Afficher la seconde image
axes[1].imshow(image2)
axes[1].axis('off')  # Masquer les axes pour la seconde image

plt.show()
```





Accuracy DENSE / Loss DENSE

In [45]:
```
# Charger les images
image1 = plt.imread(mc_28_sparse_dense_accuracy)
image2 = plt.imread(mc_28_sparse_dense_loss)

# Créer une figure et un ensemble d'axes
fig, axes = plt.subplots(1, 2, figsize=(16, 8))  # 1 ligne, 2 colonnes

# Afficher la première image
axes[0].imshow(image1)
axes[0].axis('off')  # Masquer les axes pour la première image
```

```python
# Afficher la seconde image
axes[1].imshow(image2)
axes[1].axis('off')  # Masquer les axes pour la seconde image

plt.show()
```