In [19]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [20]:
```
train_data = pd.read_csv('/content/MNIST_train.csv')
test_data = pd.read_csv('/content/MNIST_test.csv')
train_data.shape
test_data.shape
```

Out[20]:
```
(10000, 787)
```

In [21]:
```
train_data.head()
```

Out[21]:

	Unnamed: 0	index	labels	0	1	2	3	4	5	6	...	774	775	776	777	778	779	780	781	782	783
0	0	0	5	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	2	2	4	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	3	3	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	4	4	9	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

5 rows × 787 columns

In [22]:
```
X=train_data.to_numpy()
X_test=test_data.to_numpy()
```

In [23]:
```
y=X[:,2]
y_test=X_test[:,2]
```

In [24]:
```
y
y_test
```

Out[24]:
```
array([7, 2, 1, ..., 4, 5, 6])
```

In [25]:
```
set(y)
set(y_test)
```

Out[25]:
```
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
```

In [26]:
```
X=X[:,3:]
X_test=X_test[:,3:]
```

In [27]:
```
X.shape
X_test.shape
```

Out[27]:
```
(10000, 784)
```

In [28]:
```
X = X / 255
X_test = X_test/ 255
```

In [29]:
```
from scipy.stats import multivariate_normal as mvn
```

In [30]:
```
def accuracy(y,y_hat):
  return np.mean(y==y_hat)
```

In [31]:
```
class GaussBayes():
  def fit(self, X, y, epsilon=1e-1):
    self.likelihoods=dict()
    self.priors=dict()
    self.K=set(y.astype(int))

    for k in self.K:
      X_k = X[y==k, :]
      N_k, D = X_k.shape
      mu_k=X_k.mean(axis=0)

      self.likelihoods[k]={"mean":X_k.mean(axis=0), "cov":(1/(N_k-1))*np.matmul((X_k-mu_k).T, X_k-mu_k)+epsilon*
      self.priors[k]=len(X_k)/len(X)

  def predict(self,X):
    N, D = X.shape
    P_hat= np.zeros((N,len(self.K)))

    for k, l in self.likelihoods.items():
      P_hat[:,k]= mvn.logpdf(X, l["mean"],l["cov"])+np.log(self.priors[k])
```

```
    return P_hat.argmax(axis=1)
```

In [32]:
```
gb=GaussBayes()
```

In [33]:
```
gb.fit(X,y,epsilon=1e-1)
```

In [34]:
```
y_hat_bayes=gb.predict(X)
```

In [35]:
```
y_hat_bayes_test=gb.predict(X_test)
```

In [36]:
```
accuracy(y,y_hat_bayes)
```

Out[36]:
```
0.9549333333333333
```

In [37]:
```
accuracy(y_test,y_hat_bayes_test)
```

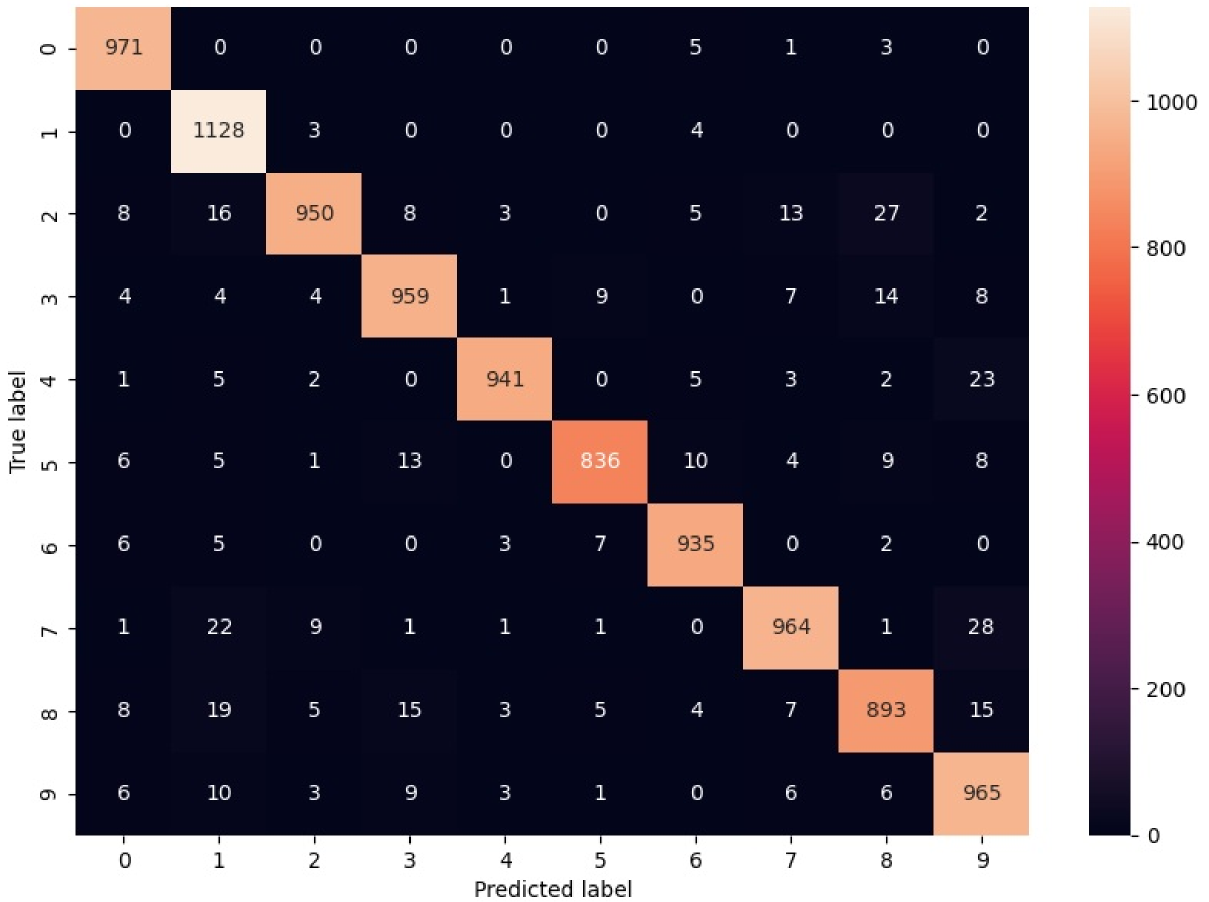
Out[37]:
```
0.9542
```

In [38]:
```
accuracy=np.mean(y_hat_bayes_test==y_test)
print(f"Accuracy:{accuracy*100:.2f}%")
```

```
Accuracy:95.42%
```

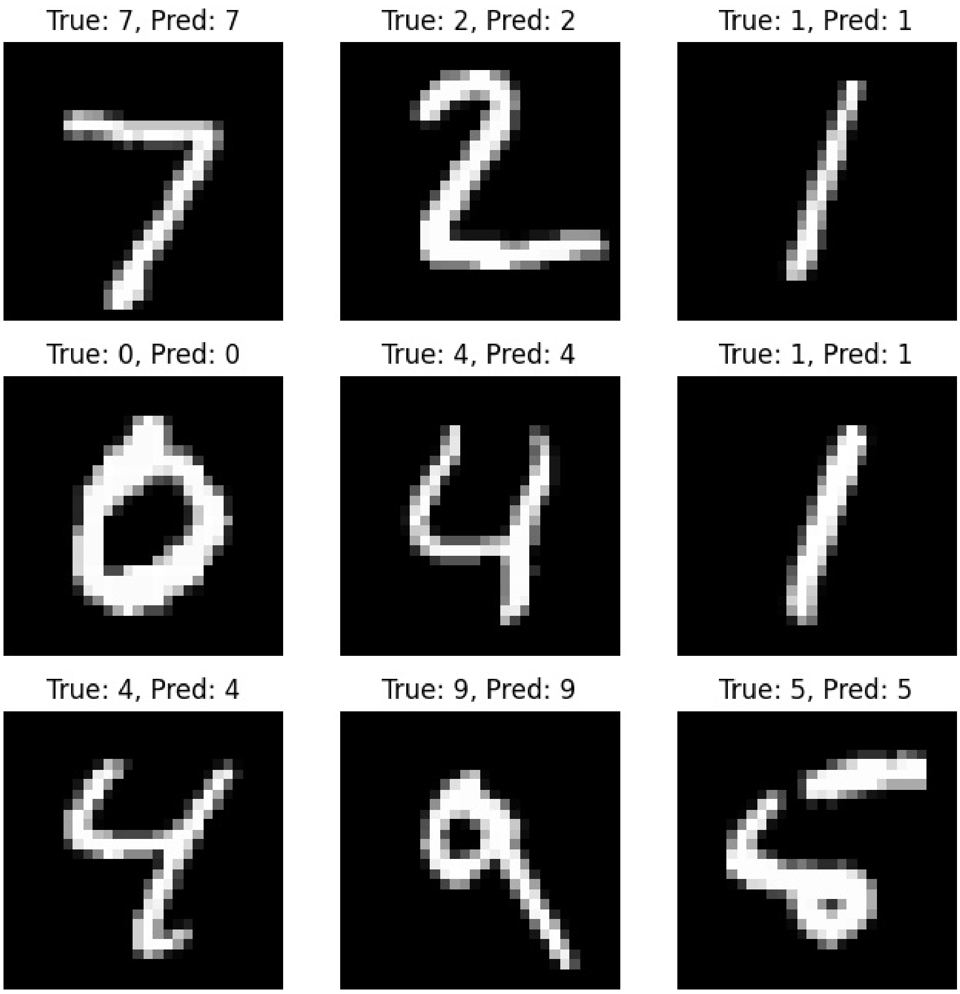
In [39]:
```
import seaborn as sns
plt.figure(figsize=(10,7))
y_actu = pd.Series(y_test, name='Actual')
y_pred = pd.Series(y_hat_bayes_test, name='Predicted')
cm = pd.crosstab(y_actu, y_pred)
ax = sns.heatmap(cm, annot=True, fmt="d")
plt.ylabel('True label')
plt.xlabel('Predicted label')
```

Out[39]:
```
Text(0.5, 47.7222222222222, 'Predicted label')
```



In [40]:
```
correct_indices = np.where(y_pred == y_test[:60000])[0]
fig, axes = plt.subplots(3, 3, figsize=(8, 8))
for i, ax in enumerate(axes.flat):
    if i < len(correct_indices):
        index = correct_indices[i]
        # Reshape the image data to 28x28
        image = X_test[index].reshape(28, 28)
        ax.imshow(image, cmap='gray')
        ax.set_title(f"True: {y_test[index]}, Pred: {y_pred[index]}")
        ax.axis('off')
    else:
```

```
        ax.axis('off')
plt.show()
```

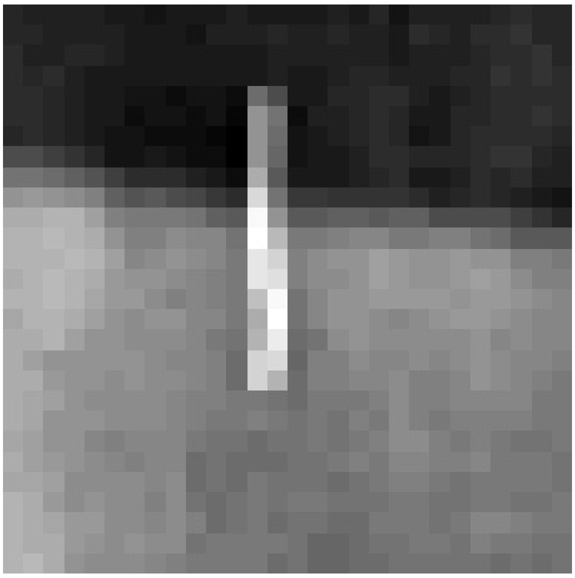


In [49]:

```
!pip install pillow
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt

def load_and_preprocess_image(filepath):
    img = Image.open(filepath).convert('L')
    img = img.resize((28, 28))
    img = np.array(img)
    img = 255 - img
    img = img.flatten()
    img = img / 255.0
    return img

image_path = '/content/Skype_Picture_2024_09_02T09_52_22_779Z.jpeg'
test_image = load_and_preprocess_image(image_path)

plt.imshow(test_image.reshape(28, 28), cmap='gray')
plt.title('Preprocessed Image')
plt.axis('off')
plt.show()
```

```
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-packages (9.4.0)
```

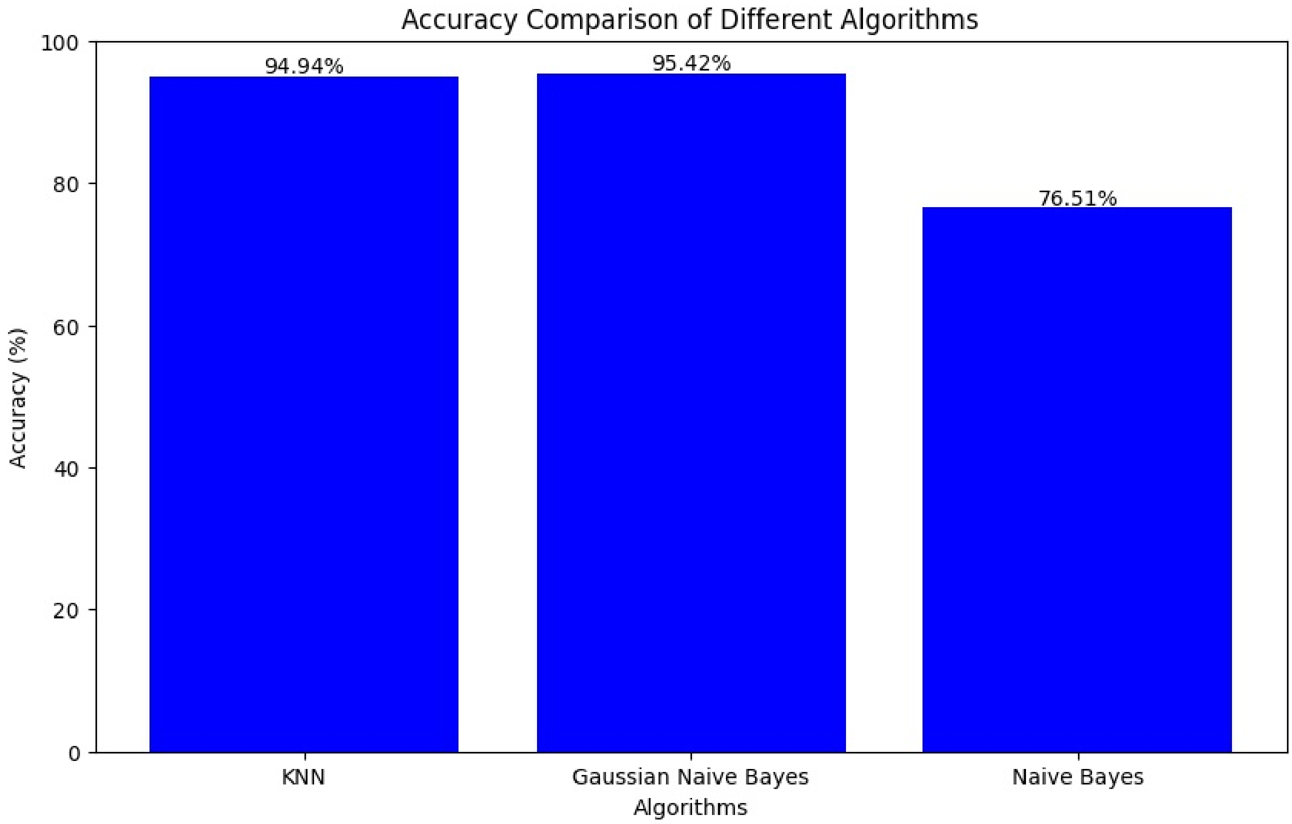
Preprocessed Image



In [50]:
```
predicted_label = gb.predict(test_image.reshape(1, -1))
print(f'Predicted Label: {predicted_label[0]}')
```

```
Predicted Label: 1
```

In [52]:
```
import matplotlib.pyplot as plt

algorithms = ['KNN', 'Gaussian Naive Bayes', 'Naive Bayes']
accuracies = [94.94, 95.42, 76.51]

plt.figure(figsize=(10, 6))
plt.bar(algorithms, accuracies, color='blue')

plt.title('Accuracy Comparison of Different Algorithms')
plt.xlabel('Algorithms')
plt.ylabel('Accuracy (%)')

for i, accuracy in enumerate(accuracies):
    plt.text(i, accuracy + 0.5, f'{accuracy:.2f}%', ha='center')

plt.ylim(0, 100)
plt.show()
```


Accuracy Comparison of Different Algorithms
100
80
60
40
20
0
Accuracy (%)
94.94%
95.42%
76.51%
KNN
Gaussian Naive Bayes
Naive Bayes
Algorithms