In [4]:
```
def open_images(filename):
    filename = "[redacted]/CNN_demo.ipynb"
    with gzip.open(filename, "rb") as file:
        data = file.read()
        return (np.frombuffer(data, dtype=np.uint8, offset=16)
                .reshape(-1, 28, 28)
                .astype(np.float32))

def open_labels(filename):
    filename = "[redacted]/CNN_demo.ipynb"
    with gzip.open(filename, "rb") as file:
        data = file.read()
        return np.frombuffer(data, dtype=np.uint8, offset=8)
```

In [5]:
```
from IPython.display import display, Markdown
```

In [6]:
```
# Part 1: Introduction
## Convolutional Neural Network for Fashion MNIST
## Convolutional Neural Network für Fashion MNIST


# In this presentation, I will showcase my ability to build and train a Convolutional Neural Network (CNN).
# The aim is to demonstrate my skills in data processing, model development, and result visualization.

# In dieser Präsentation zeige ich meine Fähigkeit, ein Convolutional Neural Network (CNN)
# dem Fashion MNIST-Datensatz zu erstellen und zu trainieren.
# Ziel ist es, meine Fähigkeiten in der Datenverarbeitung,Modellentwicklung und Ergebnisvisualisierung zu demonstrieren.
```

In [7]:
```
# Part 2: Description of the Task
# Task Context
# Kontext der Aufgabe

## Description of the Task
## Beschreibung der Aufgabe

# The task involves building a model to classify images of clothing items into one of 10 categories using the Fashion MNIST.
# This dataset is widely used for benchmarking machine learning algorithms.

# Die Aufgabe besteht darin, ein Modell zu erstellen, das Bilder von Kleidungsstücken in eine von 10 Kategorien einordnet,
# unter Verwendung des Fashion MNIST-Datensatzes.
# Dieser Datensatz wird häufig zum Benchmarking von maschinellen Lernalgorithmen verwendet.
```

In [8]:
```
# Part 3: Model Layers
# Methods and Tools
# Methoden und Werkzeuge

## List of Methods and Tools Used
## Liste der verwendeten Methoden und Werkzeuge

# 1. **TensorFlow and Keras**: For building and training the CNN model.
# 2. **NumPy**: For numerical operations and data manipulation.
# 3. **Matplotlib**: For data visualization.
# 4. **Gzip**: For decompressing the dataset files.

# 1. **TensorFlow und Keras**: Zum Erstellen und Trainieren des CNN-Modells.
# 2. **NumPy**: Für numerische Operationen und Datenmanipulation.
# 3. **Matplotlib**: Zur Datenvisualisierung.
# 4. **Gzip**: Zum Dekomprimieren der Datensatzdateien.
```

In [9]:

```
# Data Analysis
# Datenanalyse

## Loading Libraries
## Laden der Bibliotheken

import tensorflow as tf
from tensorflow import keras

print("TensorFlow version:", tf.__version__)
print("Keras version:", keras.__version__)
```

```
TensorFlow version: 2.1.0
Keras version: 2.2.4-tf
```

In [10]:

```
# Data Preprocessing
# Datenvorverarbeitung

## Functions to Open Images and Labels
## Funktionen zum Öffnen von Bildern und Labels

import gzip
import numpy as np
from tensorflow.keras.utils import to_categorical

def open_images(filename):
    with gzip.open(filename, "rb") as file:
        data = file.read()
        return (np.frombuffer(data, dtype=np.uint8, offset=16)
                .reshape(-1, 28, 28)
                .astype(np.float32))

def open_labels(filename):
    with gzip.open(filename, "rb") as file:
        data = file.read()
        return np.frombuffer(data, dtype=np.uint8, offset=8)

# Loading and Preprocessing Data
# Laden und Vorverarbeitung der Daten

X_train = open_images("[redacted]train-images-idx3-ubyte.gz")
y_train = open_labels("[redacted]train-labels-idx1-ubyte.gz")

X_test = open_images("[redacted]t10k-images-idx3-ubyte.gz")
y_test = open_labels("[redacted]/t10k-labels-idx1-ubyte.gz")

y_train = to_categorical(y_train)
y_test = to_categorical(y_test)
```

In [13]:

```
# Model Definition
# Modelldefinition

## Defining the Model Architecture
## Definition der Modellarchitektur

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, Flatten

model = Sequential()

model.add(Conv2D(10, kernel_size=(3,3), activation="relu", input_shape=(28,28,1)))
model.add(Flatten())
model.add(Dense(10, activation="softmax"))

# Model Compilation and Training
# Modellkompilierung und -training
```

```
## Compiling and Training the Model
## Kompilieren und Trainieren des Modells

model.compile(optimizer="rmsprop", loss="categorical_crossentropy", metrics=["accuracy"])

model.fit(
    X_train.reshape(60000, 28, 28, 1),
    y_train,
    epochs=10,
    batch_size=1000)
```

```
Train on 60000 samples
Epoch 1/10
60000/60000 [==============================] - 13s 219us/sample - loss: 62.9586 - accuracy: 0.5991 - lo
Epoch 2/10
60000/60000 [==============================] - 12s 200us/sample - loss: 13.7750 - accuracy: 0.7440
Epoch 3/10
60000/60000 [==============================] - 12s 201us/sample - loss: 4.5491 - accuracy: 0.8063
Epoch 4/10
60000/60000 [==============================] - 14s 225us/sample - loss: 1.2759 - accuracy: 0.8422
Epoch 5/10
60000/60000 [==============================] - 12s 197us/sample - loss: 0.4981 - accuracy: 0.8681- loss: 0.5925 - accura - ETA: 7s
Epoch 6/10
60000/60000 [==============================] - 12s 200us/sample - loss: 0.3206 - accuracy: 0.8903
Epoch 7/10
60000/60000 [==============================] - 12s 198us/sample - loss: 0.2546 - accuracy: 0.9091
Epoch 8/10
60000/60000 [==============================] - 12s 198us/sample - loss: 0.2259 - accuracy: 0.9182- los
Epoch 9/10
60000/60000 [==============================] - 12s 196us/sample - loss: 0.1996 - accuracy: 0.9273
Epoch 10/10
60000/60000 [==============================] - 12s 196us/sample - loss: 0.1804 - accuracy: 0.9339
```

Out[13]:
```
<tensorflow.python.keras.callbacks.History at 0x1c25fd485f8>
```

In [14]:
```
# Model Structure
# Modellstruktur

## Checking the Layers of the Model
## Überprüfung der Schichten des Modells

model.layers
```

Out[14]:
```
[<tensorflow.python.keras.layers.convolutional.Conv2D at 0x1c25f9a1b70>,
 <tensorflow.python.keras.layers.core.Flatten at 0x1c25f998198>,
 <tensorflow.python.keras.layers.core.Dense at 0x1c25f998828>]
```

In [22]:
```
# Weights of the First Layer
# Gewichte der ersten Schicht

## Inspecting the Weights
## Untersuchung der Gewichte

model.layers[0].weights
```

Out[22]:
```
[<tf.Variable 'conv2d/kernel:0' shape=(3, 3, 1, 10) dtype=float32, numpy=
 array([[[[-2.85780872e-03,  1.18585318e-01, -1.30591104e-02,
           -1.64691657e-01,  4.66694590e-03, -9.25477520e-02,
            1.13602601e-01,  3.65085006e-02, -9.19569582e-02,
           -1.42988488e-01]],

         [[-2.61877291e-02, -7.83542171e-02, -1.51394054e-01,
            1.06431521e-01, -1.59644946e-01,  2.77685635e-02,
            7.39484057e-02, -9.06928815e-03,  1.30068222e-02,
           -1.18249841e-01]],

         [[ 1.76664647e-02, -1.79735273e-01, -2.19657451e-01,
           -1.64638698e-01,  5.20035066e-02, -2.08313257e-01,
           -1.93146989e-01,  1.10633448e-02,  9.16855112e-02,
```

```
           9.79717523e-02]]],


        [[[ 1.01913407e-01, -2.04257518e-01, -1.06051072e-01,
           -1.83400631e-01, -1.20637543e-01,  2.15254754e-01,
           -1.91171870e-01, -1.64253756e-01,  8.06288198e-02,
           -7.68632889e-02]],

         [[-1.10321552e-01, -8.02843347e-02,  2.37416252e-02,
           -5.15200496e-02,  1.19147226e-01, -2.33394593e-01,
            4.98690344e-02,  7.45656341e-02, -1.44482823e-02,
           -8.60762671e-02]],

         [[ 3.52349132e-03,  9.18780118e-02,  8.87966305e-02,
            1.19184837e-01,  6.48180582e-03,  2.43444324e-01,
            4.98337708e-02, -7.71581940e-03, -5.12306653e-02,
            8.54695737e-02]]],


        [[[-7.64283165e-02, -1.46812975e-01,  5.36561161e-02,
           -5.87765165e-02,  1.20308802e-01, -1.60722837e-01,
           -7.40465745e-02, -3.56295183e-02, -3.09614073e-02,
            5.41790910e-02]],

         [[-2.23894138e-04,  1.41659781e-01,  1.02988914e-01,
           -1.23163722e-01, -1.79537624e-01, -1.86944887e-01,
           -1.68770686e-01, -2.64353514e-01, -7.72091374e-03,
           -3.27663809e-01]],

         [[ 1.12962071e-02, -1.48747861e-02, -1.16374061e-01,
            4.24521975e-02, -6.72580078e-02, -1.27067164e-01,
            7.30229244e-02,  7.29208579e-03, -1.51915878e-01,
           -1.68409243e-01]]]], dtype=float32)>,
 <tf.Variable 'conv2d/bias:0' shape=(10,) dtype=float32, numpy=
 array([-0.08882507, -0.04236728, -0.01803251, -0.01378375, -0.11253818,
        -0.00202528, -0.04842331, -0.1525321 , -0.06131497,  0.00882011],
       dtype=float32)>]
```

In [15]:

```
# Visualizing Weights
# Visualisierung der Gewichte

## Displaying the Weights
## Darstellung der Gewichte

import tensorflow.keras.backend as K

data = K.eval(model.layers[0].weights[0])

data[:, :, :, 0]
```

Out[15]:

```
array([[[-0.16036654],
        [ 0.06038418],
        [ 0.09603793]],

       [[-0.03488331],
        [ 0.03705579],
        [-0.09858648]],

       [[ 0.05619534],
        [-0.11196055],
        [ 0.01476851]]], dtype=float32)
```

In [16]:

```
data[:, :, :, 0].reshape(3,3)
```

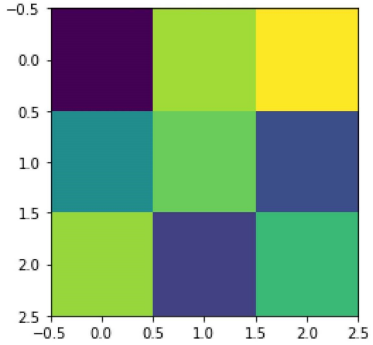
Out[16]:

```
array([[-0.16036654,  0.06038418,  0.09603793],
       [-0.03488331,  0.03705579, -0.09858648],
       [ 0.05619534, -0.11196055,  0.01476851]], dtype=float32)
```

In [17]:

```
%matplotlib inline
import matplotlib.pyplot as plt
```

```
plt.imshow(data[:, :, :, 0].reshape(3,3))
plt.show()
```

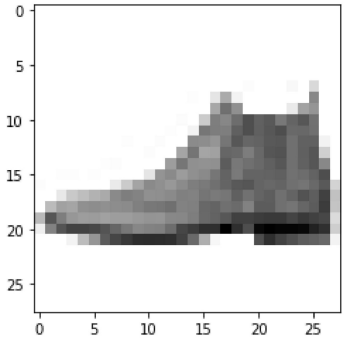


In [18]:

```
# Part 4: Visualizing Test Image
# Creating a New Model with Saved Weights
# Erstellen eines neuen Modells mit gespeicherten Gewichten

## Using Saved Weights in a New Model
## Verwendung gespeicherter Gewichte in einem neuen Modell

model12 = Sequential()

model12.add(Conv2D(10,
                   kernel_size=(3,3),
                   activation="sigmoid",
                   input_shape=(28,28,1),
                  weights=model.layers[0].get_weights()))
```

In [19]:

```
# Visualizing Test Image
# Visualisierung des Testbildes

## Displaying a Test Image
## Darstellung eines Testbildes

%matplotlib inline
import matplotlib.pyplot as plt

plt.imshow(X_test[0], cmap="gray_r")
plt.show()
```



In [20]:

```
# Prediction Using the New Model
# Vorhersage mit dem neuen Modell
```

```python
## Making Predictions
## Vorhersagen machen

result=model12.predict(X_test[0].reshape(1, 28,28, 1))
```

In [21]:

```python
result.shape
```

Out[21]:

```
(1, 26, 26, 10)
```

In [22]:

```python
result[0][:, :, 0]
```

Out[22]:

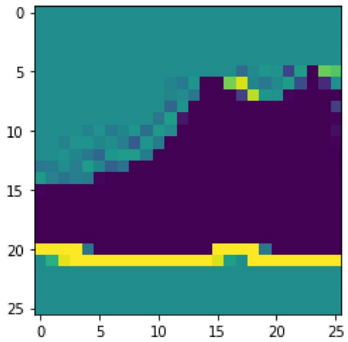
```
array([[4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01],
       [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01],
       [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01],
       [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01],
       [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01],
       [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 5.10652065e-01, 4.19992566e-01, 5.13730228e-01,
        5.13622761e-01, 5.25400281e-01, 3.13154072e-01, 7.18718052e-01,
        1.56081431e-02, 8.88700664e-01],
       [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 5.03269374e-01,
        4.78983790e-01, 4.57746893e-01, 6.24677420e-01, 1.43801168e-01,
        4.40724922e-04, 9.60509062e-01, 6.52271509e-01, 4.82685685e-01,
        4.90857542e-01, 3.33633631e-01, 5.64070761e-01, 1.05635278e-01,
        6.43383873e-06, 9.95480657e-01],
       [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.78635848e-01,
        4.31982160e-01, 5.23416042e-01, 1.92528486e-01, 2.98922885e-07,
        2.34941763e-04, 1.47755456e-03, 9.97701705e-01, 3.95941108e-01,
        4.59574103e-01, 7.30169356e-01, 3.38796377e-01, 4.51527917e-08,
        5.00127614e-01, 1.58336014e-02],
       [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
        4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
```

```
 4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 5.13705194e-01,
 4.60080832e-01, 7.17347085e-01, 2.94207293e-05, 1.97285026e-05,
 1.97317386e-05, 8.80351014e-10, 4.48833930e-07, 4.99783410e-03,
 4.21729055e-04, 3.39350966e-03, 1.80154572e-07, 1.17476117e-02,
 8.36735010e-01, 3.32843513e-07],
[4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
 4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
 4.99577284e-01, 4.99577284e-01, 5.06961107e-01, 3.71936083e-01,
 6.18163526e-01, 8.70638248e-03, 2.75523320e-07, 9.94408011e-01,
 7.35367109e-07, 2.83440034e-18, 3.73502735e-18, 2.52686566e-10,
 2.59457744e-09, 2.27670618e-10, 5.79307853e-05, 8.33501050e-04,
 4.93433490e-08, 1.02370916e-06],
[4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
 5.03269374e-01, 4.71617728e-01, 5.20998001e-01, 4.47483540e-01,
 4.99684840e-01, 5.24682403e-01, 3.69410783e-01, 6.51390851e-01,
 6.86542869e-01, 2.40976950e-09, 9.38697338e-01, 5.45311626e-03,
 3.43580636e-10, 1.12719860e-08, 1.44813317e-08, 6.32879296e-12,
 3.64544896e-12, 1.51490678e-11, 1.01908370e-07, 2.48550402e-10,
 1.65788759e-12, 2.44577647e-09],
[4.99577284e-01, 4.99577284e-01, 5.03269374e-01, 4.71617728e-01,
 4.88981277e-01, 5.08840322e-01, 4.41827834e-01, 4.93458897e-01,
 5.02475977e-01, 3.38896602e-01, 6.15416467e-01, 7.90646195e-01,
 9.09885145e-09, 8.39632750e-02, 9.99620914e-01, 8.43087022e-09,
 1.55448709e-10, 1.06621314e-06, 3.76800235e-09, 8.77977378e-12,
 2.76655698e-10, 9.87462820e-11, 1.32219999e-11, 1.06171724e-10,
 8.37795805e-12, 3.50056407e-11],
[4.99577284e-01, 4.99577284e-01, 4.74951684e-01, 5.08840322e-01,
 5.22235572e-01, 4.73486453e-01, 4.27841216e-01, 5.81257164e-01,
 3.77163827e-01, 5.59029341e-01, 8.20851266e-01, 1.22149419e-08,
 5.76568360e-04, 9.99664903e-01, 6.43718110e-08, 3.18124336e-08,
 3.53708742e-07, 1.06921391e-06, 2.83080694e-11, 6.53431920e-10,
 2.95825392e-10, 4.17290230e-11, 4.38394633e-11, 6.59781452e-10,
 2.11217545e-12, 6.82440724e-15],
[5.06961107e-01, 4.51137722e-01, 4.99396324e-01, 5.22150636e-01,
 3.71227145e-01, 4.47807223e-01, 4.94545072e-01, 4.73981440e-01,
 6.53740227e-01, 2.42961019e-01, 5.34550137e-09, 1.11385505e-03,
 9.98679340e-01, 1.02811384e-06, 7.40501410e-11, 1.00832196e-07,
 2.71171793e-05, 3.52324669e-10, 1.04248332e-10, 3.66308299e-08,
 2.74887446e-09, 5.61177459e-11, 2.60730933e-11, 4.76067630e-09,
 5.42557047e-13, 9.88092785e-17],
[4.92431164e-01, 4.68852311e-01, 4.76035267e-01, 4.42363679e-01,
 5.57152450e-01, 6.77782834e-01, 1.74611628e-01, 1.83811467e-02,
 2.43680377e-04, 1.15304530e-07, 3.35322209e-02, 9.97952938e-01,
 1.39960571e-06, 4.25615765e-10, 5.16242382e-09, 1.23339944e-07,
 9.77902005e-07, 1.94231567e-12, 1.55311264e-09, 4.02478140e-09,
 3.07592884e-09, 3.98049982e-10, 9.46832127e-11, 2.84948403e-08,
 1.53265215e-12, 5.81639283e-13],
[6.04777277e-01, 1.87484339e-01, 1.98589507e-02, 3.95269021e-02,
 2.55242530e-02, 2.74327071e-03, 1.68191564e-05, 1.09625057e-06,
 4.04572893e-06, 1.51460081e-01, 2.30227351e-01, 3.64622146e-08,
 2.23857333e-09, 2.53904602e-07, 1.71289827e-08, 2.50652334e-07,
 1.30025080e-09, 3.56946851e-12, 7.28775973e-08, 1.47677245e-10,
 2.62764477e-10, 1.58852809e-09, 6.68780586e-11, 4.21680607e-10,
 1.30825679e-08, 3.02879493e-13],
[1.26472776e-04, 5.29070567e-05, 2.32613631e-04, 3.18910975e-06,
 2.99299631e-06, 1.43086413e-06, 6.23539017e-05, 5.94997779e-04,
 2.42949856e-04, 6.05428681e-07, 5.51199930e-09, 4.89468155e-09,
 2.01413013e-06, 9.25738490e-08, 7.71102648e-09, 1.46521453e-08,
 1.09169562e-08, 2.02770991e-08, 1.42784473e-09, 1.54462272e-12,
 5.42427492e-10, 3.66819075e-10, 3.26481446e-11, 2.56562521e-10,
 1.62511601e-10, 2.60723366e-12],
[2.22110721e-08, 9.96774714e-03, 3.28251102e-04, 1.17251730e-05,
 3.49367701e-06, 3.36658413e-05, 8.86888301e-05, 1.62325165e-07,
 3.85596755e-09, 2.51345611e-09, 2.62583395e-08, 8.65637730e-08,
 4.42212169e-08, 3.24653593e-09, 2.00590189e-09, 4.66651939e-10,
 3.26209748e-09, 3.03215364e-09, 3.02775101e-12, 1.52278223e-09,
 7.00761116e-11, 2.43303468e-11, 5.01828312e-09, 1.15455125e-10,
 6.61667292e-12, 2.76703188e-10],
[1.58189319e-03, 3.24577627e-08, 1.59840052e-08, 5.54493852e-07,
 5.32064455e-08, 5.33597984e-08, 3.93916821e-07, 1.12798352e-06,
 5.64797165e-06, 4.34337466e-07, 8.68616397e-08, 7.78202214e-09,
 1.16085709e-11, 1.27274236e-09, 1.70734238e-09, 1.62602050e-11,
 2.53346602e-15, 5.29232387e-12, 3.79839875e-11, 1.31888218e-12,
 1.11328124e-14, 9.64818130e-15, 1.12492128e-13, 8.49508571e-12,
```

```
         3.96147365e-10, 3.15582727e-10],
        [2.00861394e-01, 6.61340514e-12, 1.15222810e-10, 3.24473157e-11,
         1.10014969e-10, 5.73219814e-12, 2.87562862e-10, 9.34191280e-10,
         6.97538305e-10, 1.35931810e-09, 2.38291137e-10, 3.00078851e-11,
         2.85057221e-11, 3.13092802e-10, 4.54478588e-08, 3.57871111e-09,
         4.34522661e-07, 1.57618103e-07, 1.81179668e-10, 2.06850194e-17,
         6.80262324e-16, 5.41629035e-15, 1.28634006e-14, 1.53063640e-12,
         1.34245425e-13, 4.28188951e-13],
        [1.00000000e+00, 9.99079704e-01, 1.51031269e-02, 7.38483577e-05,
         3.40691422e-08, 1.45448531e-09, 5.07189533e-11, 1.98300681e-11,
         2.13774373e-10, 2.92394109e-09, 8.34005753e-09, 5.37119513e-06,
         1.99432645e-04, 9.32966999e-04, 1.59816325e-01, 7.24581063e-01,
         8.62106800e-01, 1.26714203e-05, 4.81282818e-08, 3.55389379e-02,
         4.11912033e-08, 7.60862751e-09, 2.71958793e-08, 2.30392239e-09,
         6.04431838e-10, 4.88098673e-09],
        [4.99577284e-01, 7.59650528e-01, 9.99221563e-01, 9.99995470e-01,
         9.99988914e-01, 9.99976754e-01, 9.81671870e-01, 9.25181031e-01,
         7.36925840e-01, 3.16558033e-01, 2.34610200e-01, 5.35782613e-02,
         5.64365357e-04, 1.31905426e-06, 1.08742917e-08, 3.14820954e-06,
         2.16764316e-01, 4.99577284e-01, 1.00000000e+00, 1.00000000e+00,
         8.87764633e-01, 5.49140153e-03, 1.75460819e-02, 4.27128375e-02,
         1.25797391e-01, 2.59327095e-07],
        [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01],
        [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01],
        [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01],
        [4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01, 4.99577284e-01, 4.99577284e-01,
         4.99577284e-01, 4.99577284e-01]]], dtype=float32)
```

In [23]:

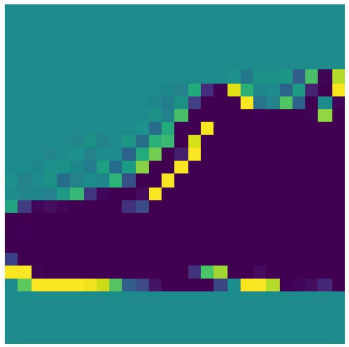
```
plt.imshow(result[0][:, :, 8])
plt.show()
```



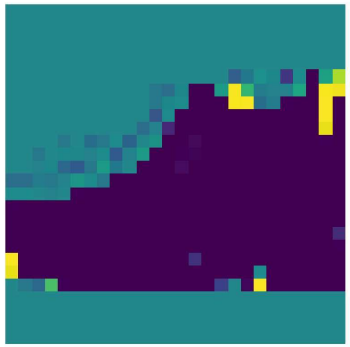
In [24]:

```
# Part 5: Visualizing All Channels

# Slide 5: Visualizing All Channels
## Visualizing All Channels
## Visualisierung aller Kanäle

## Displaying All Channels of the Convolutional Layer
## Darstellung aller Kanäle der Convolutional-Schicht


num_channels = result[0].shape[2]
height = result[0].shape[0]
width = result[0].shape[1]

num_rows = int(np.ceil(np.sqrt(num_channels)))
num_cols = int(np.ceil(num_channels / num_rows))

fig, axes = plt.subplots(num_rows, num_cols, figsize=(15, 15))

for i in range(num_channels):
    row = i // num_cols
    col = i % num_cols
    ax = axes[row, col]
    ax.imshow(result[0][:, :, i], cmap='viridis')
    ax.set_title(f'Channel {i}')
    ax.axis('off')

for j in range(i + 1, num_rows * num_cols):
    fig.delaxes(axes.flatten()[j])

plt.tight_layout()
plt.show()
```

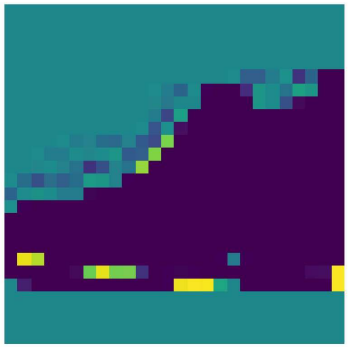
Channel 0



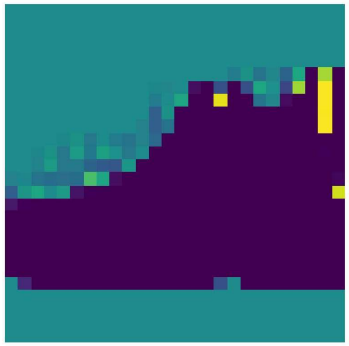
Channel 1



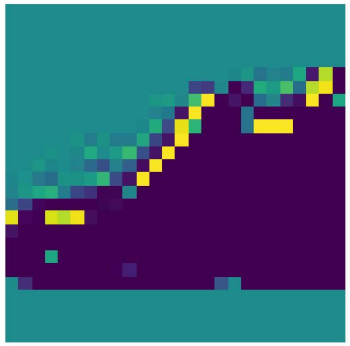
Channel 2



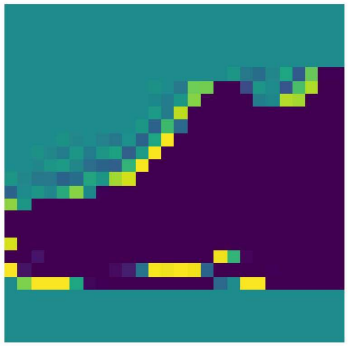
Channel 3



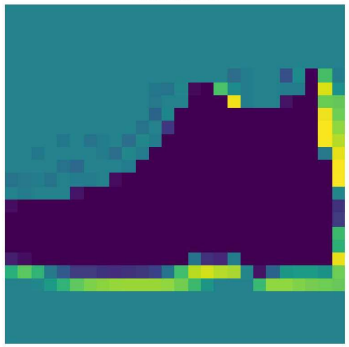
Channel 4



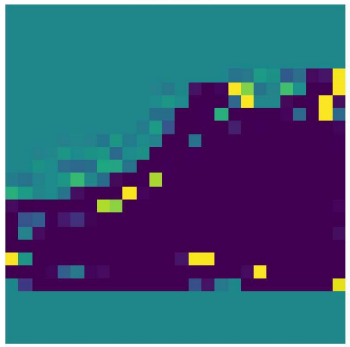
Channel 5



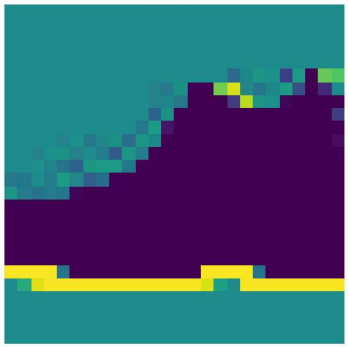
Channel 6



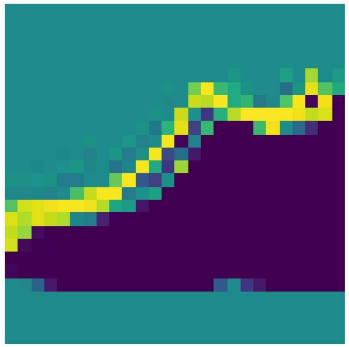
Channel 7



Channel 8



Channel 9



In [25]:

```
# Part 6: Conclusion

# Results and Conclusions
# Ergebnisse und Schlussfolgerungen

## Key Results
## Wichtige Ergebnisse


# - Successfully built and trained a CNN model to classify Fashion MNIST images.
# - Visualized the learned filters and the output of the convolutional layers.
# - Demonstrated the ability to use saved weights in a new model for predictions.

# - Erfolgreich ein CNN-Modell zur Klassifikation von Fashion MNIST-Bildern erstellt und trainiert.
# - Visualisierung der gelernten Filter und der Ausgabe der Convolutional-Schichten.
# - Fähigkeit demonstriert, gespeicherte Gewichte in einem neuen Modell für Vorhersagen zu verwenden.

# Conclusion
# Fazit

## Summary and Future Steps
## Zusammenfassung und zukünftige Schritte

# In this project, we demonstrated the process of building, training, and evaluating a CNN model for image classification.
# Future steps include experimenting with more complex architectures, tuning hyperparameters,and exploring different datasets.

# In diesem Projekt haben wir den Prozess des Erstellens, Trainings und Bewertens eines CNN-Modells zur Bildklassifikation
# demonstriert. Zukünftige Schritte umfassen Experimente mit komplexeren Architekturen, Hyperparametertuning und
# die Erforschung verschiedener Datensätze.
```

In [ ]: