**Pickle File Processing Technique**

**Creating a Model Class to Interact with Pickle File**

To demonstrate Pickle file processing, we define a Product class with the following attributes:

* ID
* name
* quantity
* unit\_price

**Example:**

import pickle

class Product:

def \_\_init\_\_(self, ID, name, quantity, unit\_price):

self.ID = ID

self.name = name

self.quantity = quantity

self.unit\_price = unit\_price

def \_\_repr\_\_(self):

return f"Product(ID={self.ID}, Name={self.name}, Quantity={self.quantity}, Unit Price={self.unit\_price})"

def get\_total\_value(self):

return self.quantity \* self.unit\_price

**Function to Save Object Model to Hard Drive**

The following function serializes a Product object and saves it to a Pickle file.

def save\_to\_pickle(product, filename):

with open(filename, 'wb') as file:

pickle.dump(product, file)

print(f"Object saved to {filename}")

**Example Usage:**

product = Product(1, "Laptop", 10, 999.99)

save\_to\_pickle(product, "product.pkl")

**Function to Load Data from Hard Drive and Restore Object**

To retrieve the saved object, we deserialize it from the Pickle file.

def load\_from\_pickle(filename):

with open(filename, 'rb') as file:

obj = pickle.load(file)

print(f"Object loaded from {filename}")

return obj

**Example Usage:**

restored\_product = load\_from\_pickle("product.pkl")

print("Restored Product Details:")

print(f"ID: {restored\_product.ID}")

print(f"Name: {restored\_product.name}")

print(f"Quantity: {restored\_product.quantity}")

print(f"Unit Price: {restored\_product.unit\_price}")

print(f"Total Value: {restored\_product.get\_total\_value()}")

**Using Pickle for the Product Class**

Pickle allows us to save and load instances of the Product class easily:

**Saving a Product Object**

product = Product(101, "Smartphone", 5, 799.99)

save\_to\_pickle(product, "product\_data.pkl")

**Loading a Product Object**

loaded\_product = load\_from\_pickle("product\_data.pkl")

print("Loaded Product Details:")

print(f"ID: {loaded\_product.ID}")

print(f"Name: {loaded\_product.name}")

print(f"Quantity: {loaded\_product.quantity}")

print(f"Unit Price: {loaded\_product.unit\_price}")

print(f"Total Value: {loaded\_product.get\_total\_value()}")

This method is particularly useful for persisting inventory data, configurations, or any structured data that needs to be stored and retrieved later. Including a method like get\_total\_value() ensures that all necessary attributes are maintained and can be used effectively after deserialization.