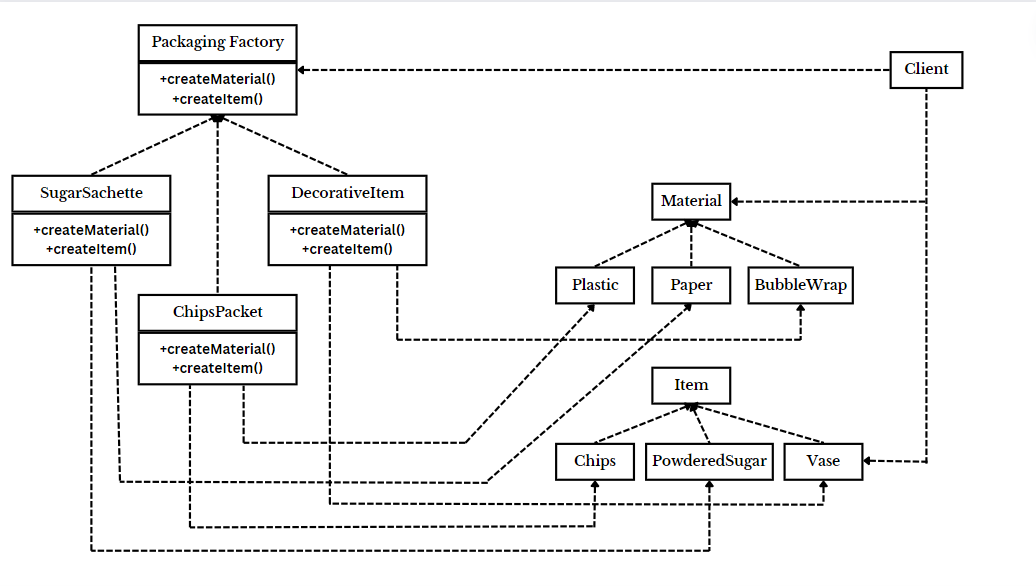
**Abstract Design Pattern**

**Assignment - 3**

Name - Heet Dobariya Roll No. - 22BCP177 Group - G5

* **Program :** Create a Builder Design Pattern for Packaging Factory example.
* **UML Diagram :**



* **Code :**

package Packaging;

interface Material

{

void displayMaterial();

void displayColour();

}

package Packaging;

class Plastic implements Material

{

@Override

public void displayMaterial()

{

System.out.println("Plastic Material: Can be used where durability is needed.");

}

@Override

public void displayColour()

{

System.out.println("Green in Colour");

}

}

package Packaging;

class Paper implements Material

{

@Override

public void displayMaterial()

{

System.out.println("Paper Material: It is easy to recycle.");

}

@Override

public void displayColour()

{

System.out.println("White Colour is the colour of paper");

}

}

package Packaging;

class BubbleWrap implements Material

{

@Override

public void displayMaterial()

{

System.out.println("Bubble Wrap: Can be used for fragile items.");

}

@Override

public void displayColour()

{

System.out.println("Transparent in colour.s");

}

}

package Packaging;

interface Item

{

void displayItem();

}

package Packaging;

class PowderedSugar implements Item

{

@Override

public void displayItem()

{

System.out.println("Powdered Sugar is the item used.");

}

}

package Packaging;

class Wafers implements Item

{

@Override

public void displayItem()

{

System.out.println("Wafers and Chips is the item used.");

}

}

package Packaging;

class Vase implements Item

{

@Override

public void displayItem()

{

System.out.println("Vase is the used.");

}

}

package Packaging;

interface PackagingFactory

{

Material createMaterial();

Item createItem();

}

package Packaging;

class SugarSachette implements PackagingFactory

{

@Override

public Material createMaterial()

{

return new Paper();

}

@Override

public Item createItem()

{

return new PowderedSugar();

}

}

package Packaging;

class ChipsPacket implements PackagingFactory

{

@Override

public Material createMaterial()

{

return new Plastic();

}

@Override

public Item createItem()

{

return new Wafers();

}

}

package Packaging;

class DecorativeItem implements PackagingFactory

{

@Override

public Material createMaterial()

{

return new BubbleWrap();

}

@Override

public Item createItem()

{

return new Vase();

}

}

package Packaging;

public class Client

{

public static void main(String[] args)

{

System.out.println("For sugar sachette:");

PackagingFactory sugarSachette = new SugarSachette();

Material material1 = sugarSachette.createMaterial();

Item item1 = sugarSachette.createItem();

material1.displayMaterial();

material1.displayColour();

item1.displayItem();

System.out.println("For chips packet:");

PackagingFactory chipsPacket = new ChipsPacket();

Material material2 = chipsPacket.createMaterial();

Item item2 = chipsPacket.createItem();

material2.displayMaterial();

material2.displayColour();

item2.displayItem();

System.out.println("For decorative item:");

PackagingFactory decorativeItem = new DecorativeItem();

Material material3 = decorativeItem.createMaterial();

Item item3 = decorativeItem.createItem();

material3.displayMaterial();

material3.displayColour();

item3.displayItem();

}

}

* **Output :**

