```
# Sweet Surrender Bakery - Ingredient Inventory & Recipe Cost Estimator

# Initial Inventory
inventory = {
    "sugar": 300,
    "butter": 30,
    "vanilla": 25,
    "baking soda": 10,
    "chocolate chips": 100,
    "cocoa powder": 800
}

# Price per unit (e.g., per gram or per unit)
price_per_unit = {
    "flour": 0.0013,
    "sugar": 0.0011,
    "eggs": 0.20,
    "butter": 0.0088,
    "milk": 0.0011,
    "vanilla": 0.17,
    "baking": 0.0026,
    "baking powder": 0.0088,
    "bananas": 0.16,
    "chocolate chips": 0.005,
    "cocoa powder": 0.0177
}

# Recipe database
recipes = {
    "cookies": {
        "flour": 280,
        "sugar": 150,
        "butter": 170,
        "eggs": 1,
        "vanilla": 5,
        "chocolate chips": 200,
        "baking powder": 2,
        "baking soda": 3
    },
    "cupcakes": {
        "flour": 250,
        "sugar": 200,
        "butter": 100,
        "eggs": 2,
        "milk": 150,
        "vanilla": 5,
        "baking": 10
    },
    "banana bread": {
        "flour": 270,
        "sugar": 150,
        "butter": 120,
        "eggs": 2,
        "vanilla": 5,
        "chocolate chips": 200,
        "baking powder": 2,
        "baking soda": 3
    }
}

# ----------- Inventory Management Functions -----------
```

```
def display_menu():
    print("\n===== INGREDIENT INVENTORY MENU =====")
    print("1. Add New Ingredient")
    print("2. View All Ingredients")
    print("3. Update Ingredient Quantity")
    print("4. Search Ingredient")
    print("5. Exit Program")
    print("6. Recipe Planning & Cost Estimation")


def add_ingredient():
    ingredient = input("Enter the ingredient name: ").strip()
    if ingredient == "":
        print("Ingredient name cannot be empty.")
        return
    if ingredient in inventory:
        print("Ingredient already exists in the inventory.")
    else:
        quantity = input("Enter quantity with unit (e.g., '10 kilos'): ").strip()
        if quantity == "":
            print("Quantity cannot be empty.")
        else:
            inventory[ingredient] = quantity
            print(f"{ingredient} added successfully with quantity: {quantity}")


def view_ingredients():
    if not inventory:
        print("Inventory is currently empty.")
    else:
        print("\n--- Current Inventory ---")
        for item, qty in inventory.items():
            print(f"{item}: {qty}")
```

```
def update_quantity():
    ingredient = input("Enter the ingredient name to update: ").strip()
    if ingredient in inventory:
        new_quantity = input("Enter the new quantity (e.g., '5 kilos'): ").strip()
        if new_quantity == "":
            print("Quantity cannot be empty.")
        else:
            inventory[ingredient] = new_quantity
            print(f"Updated {ingredient} to: {new_quantity}")
    else:
        print("Ingredient not found in the inventory.")

def search_ingredient():
    ingredient = input("Enter the ingredient name to search: ").strip()
    if ingredient in inventory:
        print(f"{ingredient}: {inventory[ingredient]}")
    else:
        print("Ingredient not found in the inventory.")

# ----------- Recipe Planning Functions -----------

def scale_recipe(recipe, num_batches):
    scaled = {}
    for ingredient, amount in recipe.items():
        scaled[ingredient] = amount * num_batches
    return scaled

def place_order(inventory, recipe):
    order = {}
    for ingredient, needed_amount in recipe.items():
        current = inventory.get(ingredient, 0)
        try:
            current = float(current)
        except:
            current = 0
        if current < needed_amount:
            order[ingredient] = needed_amount - current
    return order

def get_cost(order, prices):
    cost = 0
    for ingredient, amount in order.items():
        price = prices.get(ingredient, 0)
        cost += amount * price
    return round(cost, 2)

def recipe_planning():
    print("\nAvailable Recipes:")
    for i, recipe_name in enumerate(recipes.keys(), 1):
        print(f"{i}. {recipe_name.title()}")
    try:
        choice = int(input("Select a recipe by number: "))
        recipe_keys = list(recipes.keys())
        if choice < 1 or choice > len(recipe_keys):
            print("Invalid selection.")
            return
        selected_recipe = recipe_keys[choice - 1]
        batches = int(input("Enter number of batches: "))
        scaled = scale_recipe(recipes[selected_recipe], batches)
        print("\n--- Scaled Recipe ---")
        for item, qty in scaled.items():
            print(f"{item}: {qty}")

        order = place_order(inventory, scaled)
        if order:
            print("\n--- Ingredients to Order ---")
            for item, qty in order.items():
                print(f"{item}: {qty}")
            cost = get_cost(order, price_per_unit)
            print(f"Total cost to fulfill order: £{cost}")
        else:
            print("\nAll ingredients are in sufficient quantity.")
    except ValueError:
        print("Invalid input. Please enter numbers only.")

# ----------- Main Loop -----------

def main():
    while True:
        display_menu()
        choice = input("Enter your choice (1–6): ").strip()
        if choice == "1":
            add_ingredient()
        elif choice == "2":
            view_ingredients()
        elif choice == "3":
            update_quantity()
        elif choice == "4":
            search_ingredient()
        elif choice == "5":
            print("Thank you for using the inventory system. Goodbye!")
            break
        elif choice == "6":
            recipe_planning()
        else:
            print("Invalid option. Please enter a number between 1 and 6.")

if __name__ == "__main__":
    main()
```