Task 3

Data model

```
CREATE TABLE baskets (
    basket_id     VARCHAR PRIMARY KEY,
    customer_id   VARCHAR NOT NULL,
    total_price   NUMERIC(12, 2) NOT NULL DEFAULT 0,
    status        VARCHAR(20) NOT NULL CHECK (status IN ('NEW', ‘PENDING','EXPIRED’)),
    created_at    TIMESTAMP NOT NULL DEFAULT NOW(),
    updated_at    TIMESTAMP NOT NULL DEFAULT NOW()
);

CREATE TABLE basket_items (
    id          SERIAL PRIMARY KEY,
    basket_id    VARCHAR NOT NULL REFERENCES baskets(basket_id) ON DELETE CASCADE,
    product_id   VARCHAR NOT NULL,
    quantity     INT NOT NULL CHECK (quantity > 0),
    price        NUMERIC(12, 2) NOT NULL
);
```

Service Contract

Basket REST API:
POST /api/basket : create a new empty basket
Return: basket

PUT /api/basket/{basketId}/items: add item to basket
Body:{item_id, quantity}
Return: updated basket

GET /api/basket/{basketId} get basket
Return: basket

PUT /api/basket/{basketId}/items/{itemId}: update item quantity
Body: {quantiry}
Return basket

DELETE /api/basket/{basketId}/items/{itemId}: delete item
Return basket

DELETE /api/basket/{basketId}: delete basket

Checkout REST API:
POST /api/checkout
Body:
{
Basketid
CustomerId
ShippingAddress
PaymentInfo
}
Response: Order number

Checkout data flow:

Low-code data flow

I have limited knowledge on low-code technology, so I created a skeleton with Springboot to show the design of the API.

Checkout data flow:

- Checkout endpoint reveives checkout request from Jamstack,
- Call payment gateway and payment gateway will process payment based on the payment type in payment object, callback method waiting for the payment result.
- Once payment successful, call SAP api to create order, then return the order info to Jamstack with status