Source Code: AI Generated Report

Generated on: 2026-02-07 16:26:20

```
from fastapi import FastAPI, HTTPException

from pydantic import BaseModel

from typing import List, Dict

import pandas as pd

from fastapi.testclient import TestClient

from sqlalchemy import create_engine, Column, Float, String

from sqlalchemy.ext.declarative import declarative_base

from sqlalchemy.orm import sessionmaker


# Database setup

DATABASE_URL = "sqlite:///:memory:"

engine = create_engine(DATABASE_URL)

Base = declarative_base()


# Models

class Product(BaseModel):

    name: str

    price: float


class BillRequest(BaseModel):

    products: List[Product]


class BillResponse(BaseModel):
    total_price: float

    gst: float

    final_bill: float

    items: List[Dict[str, float]]


# Create tables

Base.metadata.create_all(bind=engine)


# FastAPI app

app = FastAPI()


@app.post("/calculate-bill", response_model=BillResponse)

async def calculate_bill(bill_request: BillRequest):

    if not bill_request.products:

        raise HTTPException(status_code=400, detail="No products provided")


    # Extract product prices into a DataFrame

    product_data = [{"name": product.name, "price": product.price} for product in
bill_request.products]

    df = pd.DataFrame(product_data)


    # Calculate total price, GST, and final bill

    total_price = df['price'].sum()

    gst = total_price * 0.12

    final_bill = total_price + gst
    # Prepare response
    response = BillResponse(
        total_price=total_price,
        gst=gst,
        final_bill=final_bill,
        items=[{"name": item["name"], "price": item["price"]} for item in product_data]
    )
    return response


# Test client
client = TestClient(app)


def test_calculate_bill():
    response = client.post("/calculate-bill", json={
        "products": [
            {"name": "Product A", "price": 100.0},
            {"name": "Product B", "price": 200.0}
        ]
    })
    print(response.json())


test_calculate_bill()
```