Source Code: AI Generated Report

Generated on: 2026-02-07 16:22:56

```
from fastapi import FastAPI, HTTPException

from pydantic import BaseModel

import pandas as pd

from fastapi.testclient import TestClient

from sqlalchemy import create_engine

from sqlalchemy.ext.declarative import declarative_base

from sqlalchemy.orm import sessionmaker


# Database setup

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

engine = create_engine(DATABASE_URL)

Base = sqlalchemy.orm.declarative_base()  # Updated line

SessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=engine)


# FastAPI app

app = FastAPI()


# Data models

class Product(BaseModel):

    name: str

    price: float


class BillRequest(BaseModel):
    products: list[Product]


# API endpoint
@app.post("/calculate-bill")
async def calculate_bill(bill_request: BillRequest):
    try:
        # Convert products to a DataFrame
           df = pd.DataFrame([product.dict() for product in bill_request.products])  #
Updated line

        # Calculate total price
        total_price = df['price'].sum()

        # Calculate GST
        gst = total_price * 0.12

        # Calculate final bill
        final_bill = total_price + gst

        return {"total": total_price, "gst": gst, "final_bill": final_bill}
    except ValueError as e:
        raise HTTPException(status_code=400, detail=str(e))

# Create database tables
Base.metadata.create_all(bind=engine)
```

```
# Testing

client = TestClient(app)


def test_calculate_bill():

    response = client.post("/calculate-bill", json={

        "products": [

            {"name": "Product A", "price": 100.00},

            {"name": "Product B", "price": 200.00},

            {"name": "Product C", "price": 150.00}

        ]

    })

    print(response.json())

    assert response.status_code == 200

    assert response.json() == {

        "total": 378.00,

        "gst": 18.00,

        "final_bill": 396.00

    }


# Run the test

test_calculate_bill()
```