# Source Code: AI Generated Report

*Generated on: 2026-02-07 16:21:51*

```
# main.py
from fastapi import FastAPI, HTTPException
from pydantic import BaseModel, conlist
import pandas as pd
from typing import List
from sqlalchemy import create_engine, Column, Float, String
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker

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

engine = create_engine(DATABASE_URL)
SessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=engine)
Base = declarative_base()

app = FastAPI()

class Product(BaseModel):
    name: str
    price: float

class BillRequest(BaseModel):
    products: List[Product]
```

```
class ProductModel(Base):
    __tablename__ = "products"
    id = Column(String, primary_key=True)
    name = Column(String)
    price = Column(Float)

Base.metadata.create_all(bind=engine)

@app.post("/calculate-bill")
def calculate_bill(bill_request: BillRequest):
    try:
        df = pd.DataFrame([{"name": product.name, "price": product.price} for product in
bill_request.products])

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

        gst = total_price * 0.12

        final_bill = total_price + gst

        return {
            "total_price": total_price,
            "gst": gst,
            "final_bill": final_bill
        }
```

```
    except Exception as e:
        raise HTTPException(status_code=400, detail=str(e))


# test_main.py
from fastapi.testclient import TestClient
from main import app

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())
    assert response.status_code == 200
    assert response.json() == {
        "total_price": 300.0,
        "gst": 36.0,
        "final_bill": 336.0
    }

def test_calculate_bill_empty():
    response = client.post("/calculate-bill", json={"products": []})
```

```
    print(response.json())
    assert response.status_code == 422


def test_calculate_bill_invalid_price():
    response = client.post("/calculate-bill", json={
        "products": [
            {"name": "Product A", "price": "invalid_price"}
        ]
    })
    print(response.json())
    assert response.status_code == 422
```