اول حاجه يا شباب لازم نعملها اننا نشغل المشروع لوكال

نزّل Docker Desktop (غالبًا Compose بييجي مع)
ثبّت Python 3.11 (Python 3.11 عشان كل الخدمات مطلوب تكون)

2-اعمل فولدر المشروع بالطريقه اللي تعجبك
وبعدين هنعمل
`docker-compose.yml` يشغل (Kafka + DBs + Services)

هنا انا في كود مش متاكد هيشتغل صح ول الا بس دا
لوكال (علشان سهل) 1 Kafka broker
services:

zookeeper:

image: confluentinc/cp-zookeeper:7.6.1

environment:

ZOOKEEPER_CLIENT_PORT: 2181

ports: ["2181:2181"]

kafka:

image: confluentinc/cp-kafka:7.6.1

depends_on: [zookeeper]

ports: ["9092:9092"]

environment:

KAFKA_BROKER_ID: 1

KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181

KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://kafka:9092

KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: 1

docdb:

image: postgres:16

environment:

POSTGRES_USER: doc

POSTGRES_PASSWORD: doc

POSTGRES_DB: documentdb

ports: ["5433:5432"]

```
quizdb:
image: postgres:16
environment:
POSTGRES_USER: quiz
POSTGRES_PASSWORD: quiz
POSTGRES_DB: quizdb
ports: ["5434:5432"]

document-reader:
build: ./services/document-reader
depends_on: [kafka, docdb]
environment:
KAFKA_BOOTSTRAP: kafka:9092
DOC_DB_URL: postgresql://doc:doc@docdb:5432/documentdb
ports: ["8001:8000"]

quiz:
build: ./services/quiz
depends_on: [kafka, quizdb]
environment:
KAFKA_BOOTSTRAP: kafka:9092
QUIZ_DB_URL: postgresql://quiz:quiz@quizdb:5432/quizdb
ports: ["8002:8000"]

api-gateway:
build: ./api-gateway
depends_on: [document-reader, quiz]
environment:
DOC_URL: http://document-reader:8000
QUIZ_URL: http://quiz:8000
```

ports: ["8000:8000"]

zookeeper و3 brokers 3 هتشتغل ب AWS لكن انت ع

3service- المفروض تكتب بقي كود ال
Document Reader (FastAPI + Kafka Producer)
انت هتكون عملت الفولدرات
services/document-reader/
جواه نعمل
services/document-reader/requirements.txt
وبرده
fastapi==0.115.0

uvicorn[standard]==0.30.6

httpx==0.27.2

confluent-kafka==2.5.0

python-multipart==0.0.9

واعمل فولدر
services/document-reader/Dockerfile

FROM python:3.11-slim

WORKDIR /app

COPY requirements.txt .

RUN pip install --no-cache-dir -r requirements.txt

COPY main.py .

EXPOSE 8000

CMD ["uvicorn", "main:app", "--host=0.0.0.0", "--port=8000"]

services/document-reader/main.py

import json, os, uuid

from fastapi import FastAPI, UploadFile, File

from confluent_kafka import Producer

```
app = FastAPI(title="Document Reader Service")

KAFKA_BOOTSTRAP = os.getenv("KAFKA_BOOTSTRAP", "kafka:9092")
producer = Producer({"bootstrap.servers": KAFKA_BOOTSTRAP})

TOPIC_DOCUMENT_UPLOADED = "document.uploaded"
TOPIC_DOCUMENT_PROCESSED = "document.processed"
TOPIC_NOTES_GENERATED = "notes.generated"

def publish(topic: str, payload: dict):
producer.produce(topic, json.dumps(payload).encode("utf-8"))
producer.flush(5)

@app.get("/health")
def health():
return {"ok": True}

# مطلوب endpoint upload: POST /api/documents/upload :contentReference[oaicite:11]{index=11}
@app.post("/api/documents/upload")
async def upload_document(file: UploadFile = File(...)):
doc_id = str(uuid.uuid4())
content = (await file.read()).decode("utf-8", errors="ignore")

# 1) event: document.uploaded :contentReference[oaicite:12]{index=12}
publish(TOPIC_DOCUMENT_UPLOADED, {
"document_id": doc_id,
"filename": file.filename
})

# (MVP) “processing” بسيط: نعتبر القراءة تمت فورًا
publish(TOPIC_DOCUMENT_PROCESSED, {
"document_id": doc_id,
```

"text": content[:5000] # خلیه محدود

})

(MVP) notes generated

notes = content[:300] if content else ""

publish(TOPIC_NOTES_GENERATED, {

"document_id": doc_id,

"notes": notes

})

return {"document_id": doc_id, "message": "uploaded & processed (MVP)"}

5) کود اکتبservices Quiz (FastAPI + Kafka Consumer/Producer)

services/quiz/

services/quiz/requirements.txt

fastapi==0.115.0

uvicorn[standard]==0.30.6

confluent-kafka==2.5.0

services/quiz/Dockerfile

FROM python:3.11-slim

WORKDIR /app

COPY requirements.txt .

RUN pip install --no-cache-dir -r requirements.txt

COPY main.py .

EXPOSE 8000

```
CMD ["uvicorn", "main:app", "--host=0.0.0.0", "--port=8000"]
```

services/quiz/main.py

```
import json, os, threading, uuid
from fastapi import FastAPI
from confluent_kafka import Consumer, Producer

app = FastAPI(title="Quiz Service")

KAFKA_BOOTSTRAP = os.getenv("KAFKA_BOOTSTRAP", "kafka:9092")
producer = Producer({"bootstrap.servers": KAFKA_BOOTSTRAP})

TOPIC_QUIZ_REQUESTED = "quiz.requested"
TOPIC_QUIZ_GENERATED = "quiz.generated"
TOPIC_NOTES_GENERATED = "notes.generated"

def publish(topic: str, payload: dict):
producer.produce(topic, json.dumps(payload).encode("utf-8"))
producer.flush(5)

def kafka_worker():
    c = Consumer({
"bootstrap.servers": KAFKA_BOOTSTRAP,
"group.id": "quiz-service",
"auto.offset.reset": "earliest",
})
```

```
c.subscribe([TOPIC_QUIZ_REQUESTED, TOPIC_NOTES_GENERATED])

while True:
msg = c.poll(1.0)
if msg is None:
continue
if msg.error():
continue

event = json.loads(msg.value().decode("utf-8"))

# لما ييجي notes.generated (مذكور انه quiz يستهلك notes.generated)
:contentReference[oaicite:15]{index=15}
if msg.topic() == TOPIC_NOTES_GENERATED:
doc_id = event["document_id"]
quiz_id = str(uuid.uuid4())
quiz = {
"quiz_id": quiz_id,
"document_id": doc_id,
"questions": [
{"q": "What is the main topic of the notes?", "type": "short"},
{"q": "True/False: Document was processed successfully.", "type": "tf"}
]
}
publish(TOPIC_QUIZ_GENERATED, quiz)
```

لو user طلب generate quiz (topic quiz.requested) :contentReference[oaicite:16]{index=16}

if msg.topic() == TOPIC_QUIZ_REQUESTED:

doc_id = event["document_id"]

quiz_id = str(uuid.uuid4())

publish(TOPIC_QUIZ_GENERATED, {

"quiz_id": quiz_id,

"document_id": doc_id,

"questions": [{"q": "Sample MCQ?", "type": "mcq"}]

})

@app.on_event("startup")

def start_consumer():

threading.Thread(target=kafka_worker, daemon=True).start()

@app.get("/health")

def health():

return {"ok": True}

مطلوب: POST /api/quiz/generate :contentReference[oaicite:17]{index=17}

@app.post("/api/quiz/generate")

def generate_quiz(document_id: str):

publish(TOPIC_QUIZ_REQUESTED, {"document_id": document_id})

return {"message": "quiz requested", "document_id": document_id}

6) اكتب API Gateway (Routing بسيط)

جوه

اعمل /api-gateway:

شغّل كل حاجة لوكال

docker compose up –build

لو كل حاجه تمام:

- Gateway: http://localhost:8000/health
- Doc service: http://localhost:8001/health
- Quiz service: http://localhost:8002/health