Albatta! Men siz uchun **FastAPI + Gradio + Docker + AWS EC2 + ECR** butun jarayonini **bosqichma-bosqich** ketma-ket, kodlar bilan yozaman.

Har bir qadamda **qayerda bajarilishi kerakligini** aniqlaymiz: lokal mashina yoki EC2 (`[local]` yoki `[AWS]`).

1️⃣ LOKAL: FastAPI + Gradio loyihasini tayyorlash

1.1 Loyiha tuzilishi:

```
Hotel-Booking-Prediction/
├─ app/
│  ├─ __init__.py
│  └─ main.py
├─ Models/
│  └─ Advanced/
│     └─ Stacking_Classifier_best_model.joblib
├─ requirements.txt
└─ Dockerfile
```

1.2 requirements.txt `[local]`

```
fastapi uvicorn pandas scikit-learn gradio joblib
```

1.3 app/main.py `[local]`

```
import joblib import pandas as pd import gradio as gr MODEL_PATH =
r"../Models/Advanced/Stacking_Classifier_best_model.joblib" pipeline =
joblib.load(MODEL_PATH) def predict( hotel, lead_time, arrival_date_year,
arrival_date_month, arrival_date_week_number, arrival_date_day_of_month,
stays_in_weekend_nights, stays_in_week_nights, adults, children, babies, meal, country,
market_segment, distribution_channel, is_repeated_guest, previous_cancellations,
previous_bookings_not_canceled, reserved_room_type, assigned_room_type, booking_changes,
deposit_type, agent, company, days_in_waiting_list, customer_type, adr,
required_car_parking_spaces, total_of_special_requests, reservation_status,
reservation_status_date, city ): df = pd.DataFrame([{ "hotel": hotel, "lead_time":
lead_time, "arrival_date_year": arrival_date_year, "arrival_date_month":
arrival_date_month, "arrival_date_week_number": arrival_date_week_number,
```

```
"arrival_date_day_of_month": arrival_date_day_of_month, "stays_in_weekend_nights":
stays_in_weekend_nights, "stays_in_week_nights": stays_in_week_nights, "adults": adults,
"children": children, "babies": babies, "meal": meal, "country": country,
"market_segment": market_segment, "distribution_channel": distribution_channel,
"is_repeated_guest": is_repeated_guest, "previous_cancellations": previous_cancellations,
"previous_bookings_not_canceled": previous_bookings_not_canceled, "reserved_room_type":
reserved_room_type, "assigned_room_type": assigned_room_type, "booking_changes":
booking_changes, "deposit_type": deposit_type, "agent": agent, "company": company,
"days_in_waiting_list": days_in_waiting_list, "customer_type": customer_type, "adr": adr,
"required_car_parking_spaces": required_car_parking_spaces, "total_of_special_requests":
total_of_special_requests, "reservation_status": reservation_status,
"reservation_status_date": reservation_status_date, "city": city }]) pred =
pipeline.predict(df)[0] proba = pipeline.predict_proba(df)[0][1] return {"prediction":
int(pred), "probability": float(proba)} demo = gr.Interface( fn=predict, inputs=
[gr.Textbox(label="Hotel"), gr.Number(label="Lead Time"), gr.Number(label="Arrival Year"),
gr.Textbox(label="Arrival Month"), gr.Number(label="Arrival Week Number"),
gr.Number(label="Arrival Day of Month"), gr.Number(label="Stays in Weekend Nights"),
gr.Number(label="Stays in Week Nights"), gr.Number(label="Adults"),
gr.Number(label="Children"), gr.Number(label="Babies"), gr.Textbox(label="Meal"),
gr.Textbox(label="Country"), gr.Textbox(label="Market Segment"),
gr.Textbox(label="Distribution Channel"), gr.Number(label="Is Repeated Guest (0 or 1)"),
gr.Number(label="Previous Cancellations"), gr.Number(label="Previous Bookings Not
Canceled"), gr.Textbox(label="Reserved Room Type"), gr.Textbox(label="Assigned Room
Type"), gr.Number(label="Booking Changes"), gr.Textbox(label="Deposit Type"),
gr.Number(label="Agent"), gr.Number(label="Company"), gr.Number(label="Days in Waiting
List"), gr.Textbox(label="Customer Type"), gr.Number(label="ADR"),
gr.Number(label="Required Car Parking Spaces"), gr.Number(label="Total of Special
Requests"), gr.Textbox(label="Reservation Status"), gr.Textbox(label="Reservation Status
Date"), gr.Textbox(label="City")], outputs="json", title="Hotel Booking Cancellation
Prediction", description="Enter booking details to predict cancellation" ) if __name__ ==
"__main__": demo.launch(server_name="0.0.0.0", server_port=8000)
```

1.4 Dockerfile `[local]`

```
FROM python:3.11-slim WORKDIR /app COPY requirements.txt . RUN pip install --no-cache-dir
-r requirements.txt COPY app ./app COPY Models ./Models CMD ["uvicorn", "app.main:app", "-
-host", "0.0.0.0", "--port", "8000"]
```

2️⃣ LOKAL: Docker image yaratish

```
docker build -t hotel-booking-cancel:latest .
```

Tekshirish:

```
docker images
```

---

3 LOKAL: Docker container run

```
docker run -d \ --name hotel-api \ --restart unless-stopped \ -p 8000:8000 \ hotel-booking-cancel:latest
```

- Tekshirish: `docker ps`
- Browser: `http://localhost:8000`

---

4 AWS: EC2 tayyorlash `[AWS]`

1. **EC2 instance yaratish** (Amazon Linux 2023, t2.micro yoki t3.micro)
2. **Security group**: 22 (SSH), 8000 (HTTP) port ochiq bo'lishi kerak
3. **Key pair**: `.pem` faylini saqlang

---

5 AWS: EC2 ga SSH ulanish `[local]`

```
chmod 400 ~/Desktop/new_key_pairs.pem ssh -i ~/Desktop/new_key_pairs.pem ec2-user@13.204.47.155
```

---

6 AWS EC2: Docker o'rnatish va ishga tushirish `[AWS]`

```
sudo systemctl enable docker sudo systemctl start docker sudo usermod -aG docker ec2-user newgrp docker docker --version docker run hello-world
```

---

7 AWS EC2: Loyiha fayllarini ko'chirish `[local → AWS]`

```
scp -i ~/Desktop/new_key_pairs.pem -r ~/Desktop/Hotel\ Booking\ Cancellation\ Prediction
ec2-user@13.204.47.155:~/hotel_app
```

---

8 AWS EC2: Docker image build [AWS]

```
cd ~/hotel_app docker build -t hotel-booking-cancel:latest . docker images
```

---

9 AWS ECR: repository yaratish va login [AWS]

```
aws ecr create-repository \ --repository-name hotel-booking-cancel \ --region ap-south-1
aws ecr get-login-password --region ap-south-1 | \ docker login --username AWS --password-
stdin 221671810317.dkr.ecr.ap-south-1.amazonaws.com
```

---

10 AWS EC2: Docker image tag va push [AWS]

```
docker tag hotel-booking-cancel:latest 221671810317.dkr.ecr.ap-south-
1.amazonaws.com/hotel-booking-cancel:latest docker push 221671810317.dkr.ecr.ap-south-
1.amazonaws.com/hotel-booking-cancel:latest
```

---

11 AWS EC2: Docker container run [AWS]

1 Agar eski container bo‘lsa, to‘xtatish va o‘chirish:

```
docker stop hotel-api docker rm hotel-api
```

2 Yangi container run:

```
docker run -d \ --name hotel-api \ --restart unless-stopped \ -p 8000:8000 \
221671810317.dkr.ecr.ap-south-1.amazonaws.com/hotel-booking-cancel:latest \ uvicorn
app.main:app --host 0.0.0.0 --port 8000
```

3 Tekshirish:

```
docker ps
```

- `PORTS` qatorida `0.0.0.0:8000->8000/tcp` bo‘lishi kerak
- Browser orqali: `http://13.204.47.155:8000`

✅ Shu bilan FastAPI + Gradio + Docker + AWS EC2 + ECR butun jarayoni tayyor