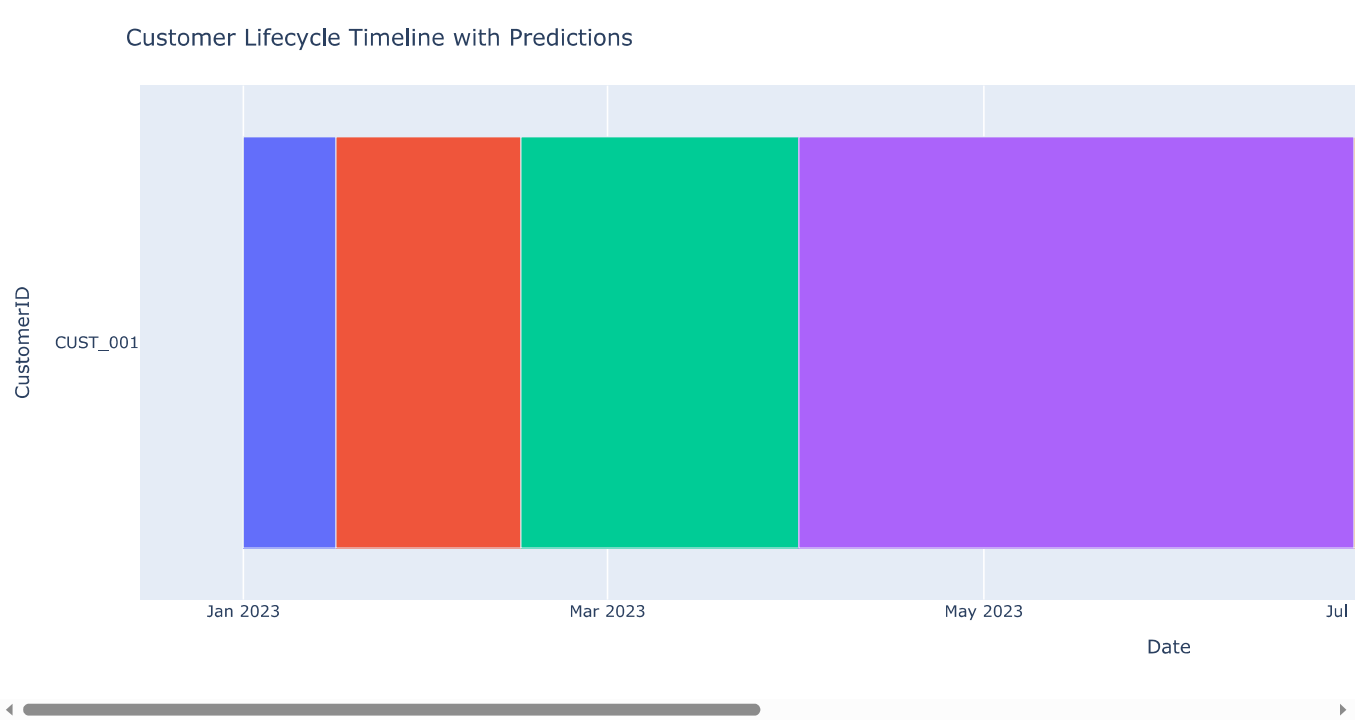
```
pip install pandas plotly
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: plotly in /usr/local/lib/python3.11/dist-packages (5.24.1)
Requirement already satisfied: numpy>=1.23.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.11/dist-packages (from plotly) (9.1.2)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from plotly) (24.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
```

```
import pandas as pd
import numpy as np
import plotly.express as px
from datetime import timedelta, datetime

# Step 1: Generate synthetic customer lifecycle data
np.random.seed(42)

customer_id = "CUST_001"
start_date = datetime(2023, 1, 1)

# Define lifecycle stages
stages = [
    ("Awareness", 15),
    ("acquisition", 30),
    ("conversion", 45),
    ("Retention", 90),
    ("Loyality", 120),
]

# Generate timeline dates
dates = [start_date]
events = []
for stage, duration in stages:
    events.append(stage)
    dates.append(dates[-1] + timedelta(days=duration))

# create timeline dates
timeline_df = pd.DataFrame({
    "CustomerID": customer_id,
    "Stage": events,
    "StartDate": dates[:-1],
    "EndDate": dates[1:]
})

# Add Predictions
timeline_df["PredictedNextStage"] = timeline_df["Stage"].shift(-1)
timeline_df["PredictionScore"] = np.round(np.random.uniform(0.7, 0.95, size=len(timeline_df)), 2)

# Step 3: Visualize using a timeline plot
fig = px.timeline(
    timeline_df,
    x_start="StartDate",
    x_end="EndDate",
    y="CustomerID",
    color="Stage",
    hover_data=["Stage", "PredictedNextStage", "PredictionScore"]
)

fig.update_yaxes(autorange="reversed")
fig.update_layout(title="Customer Lifecycle Timeline with Predictions", xaxis_title="Date")

fig.show()
```


Customer Lifecycle Timeline with Predictions
CustomerID
CUST_001
Jan 2023
Mar 2023
May 2023
Jul
Date