**Introduction**

This report details the design, implementation, and justification of a dynamic pricing engine for urban parking lots, using only Python, pandas, numpy, and Pathway, as required by the project guidelines. The report covers the data preparation, feature engineering, three pricing models (baseline, demand-based, competitive), and the logic behind each step. It also explains the demand function, key assumptions, and how price changes are governed by demand and competition.

**1. Data Preparation**

**1.1 Data Loading**

* The dataset contains real-time operational data from 14 parking lots over 73 days, sampled at 18 time points per day.
* Each record includes location, lot features, vehicle information, environmental conditions, and timestamps.

**1.2 Data Cleaning**

* **Missing Values:** All rows with missing values are dropped to ensure model reliability.
* **Categorical Encoding:**
  + *Traffic Condition* mapped to numeric (low=0, average=1, high=2).
  + *Vehicle Type* mapped to weights (car=1.0, bike=0.5, truck=2.0, cycle=0.7).
  + *Special Day* converted to integer (0 or 1).
* **Timestamp Creation:** Date and time columns are merged into a single datetime field for sorting and time-based analysis.

**Justification:**  
Cleaning and standardizing the data ensures that the models receive consistent, meaningful input, which is crucial for accurate, explainable pricing.

**2. Feature Engineering**

* **Occupancy Rate:** Calculated as Occupancy / Capacity for each lot and time point.
* **Normalized Queue Length:** Queue length divided by its maximum value for scaling.
* **Other Features:** Encoded traffic level, vehicle type weight, and special day indicator are included as direct features.

**Justification:**  
These features are selected based on their direct impact on parking demand and supply, as well as their use in real-world dynamic pricing systems.

**3. Pricing Models**

**3.1 Baseline Linear Model**

**Formula:**

Pricet+1=Pricet+α×OccupancyRatePrice*t*+1=Price*t*+*α*×OccupancyRate

* Starts from a base price of $10.
* At each time step, the price increases linearly with occupancy rate.

**Justification:**  
This model provides a simple, explainable reference for how prices might respond to increasing demand.

**3.2 Demand-Based Model**

**Demand Function:**

Demand=α⋅OccupancyRate+β⋅QueueLength−γ⋅TrafficLevel+δ⋅IsSpecialDay+ϵ⋅VehicleTypeWeightDemand=*α*⋅OccupancyRate+*β*⋅QueueLength−*γ*⋅TrafficLevel+*δ*⋅IsSpecialDay+*ϵ*⋅VehicleTypeWeight

**Price Update:**

Pricet=BasePrice⋅(1+λ⋅NormalizedDemand)Price*t*=BasePrice⋅(1+*λ*⋅NormalizedDemand)

* Demand is normalized per lot.
* Price is bounded between 0.5x and 2x the base price for stability.

**Justification:**  
This model captures the multifaceted drivers of parking demand, allowing for more nuanced, real-time price adjustments. Normalization ensures prices remain stable and interpretable.

**3.3 Competitive Pricing Model**

* **Proximity Calculation:** Uses latitude and longitude to find nearby lots (within ~1km).
* **Competitive Logic:**
  + If the lot is full and nearby lots are cheaper, reduce price to attract rerouting.
  + If all nearby lots are more expensive, increase price (but not above double base price).
* **Implementation:** At each time and lot, competitor prices are compared and the price is adjusted up or down by up to 5%.

**Justification:**  
Urban parking is a competitive market. Factoring in competitor prices ensures the model adapts to local supply and demand, maximizing both occupancy and revenue.

**4. Real-Time Processing with Pathway**

* **Streaming Ingestion:** Pathway processes the dataset as a real-time stream, preserving timestamp order.
* **Feature Computation:** All features and pricing logic are computed in real time as new data arrives.
* **Continuous Output:** Updated prices are emitted for each lot and time point, ready for visualization or deployment.

**Justification:**  
Real-time processing is essential for urban parking, where demand can change rapidly due to events, traffic, or competition.

**5. Visualization**

* **Bokeh Plots:** Real-time line charts show price evolution for each lot, comparing baseline, demand-based, and competitive prices.
* **Purpose:** Visualizations help operators and stakeholders understand and trust the pricing engine’s decisions.

**Demand Function: Explanation**

The demand function is a weighted sum of key features:

* **Occupancy Rate (α*α*)**: Directly increases demand as more spots are filled.
* **Queue Length (β*β*)**: Longer queues indicate higher demand.
* **Traffic Level (−γ−*γ*)**: High traffic may reduce demand (harder to access).
* **Special Day (δ*δ*)**: Events or holidays boost demand.
* **Vehicle Type Weight (ϵ*ϵ*)**: Larger vehicles may be charged more due to space usage.

**Coefficients** are set based on intuition and can be tuned empirically.

**Assumptions**

* **Data Quality:** All input data is accurate and timely.
* **Feature Impact:** The chosen features are sufficient to model demand.
* **Competitor Proximity:** Euclidean distance is a reasonable proxy for lot proximity.
* **Price Bounds:** Prices should not be erratic—bounded between 0.5x and 2x base price.
* **Vehicle Type:** Each vehicle type’s impact on demand is proportional to its assigned weight.

**Price Dynamics: Demand and Competition**

* **With Demand:**
  + As occupancy, queue, or special days increase, demand rises, and so does price.
  + If traffic is high, price may decrease slightly to attract users.
  + Vehicle type influences price (e.g., trucks pay more).
* **With Competition:**
  + If a lot is full and cheaper alternatives exist nearby, price is reduced to encourage rerouting.
  + If all competitors are more expensive, price is increased (but not excessively) to maximize revenue while remaining attractive.

**Conclusion**

This system provides a robust, explainable, and scalable approach to dynamic pricing for urban parking, using only the permitted tools and libraries. Each model builds on the previous, adding sophistication and real-world realism, ensuring prices reflect both demand and local competition in real time.