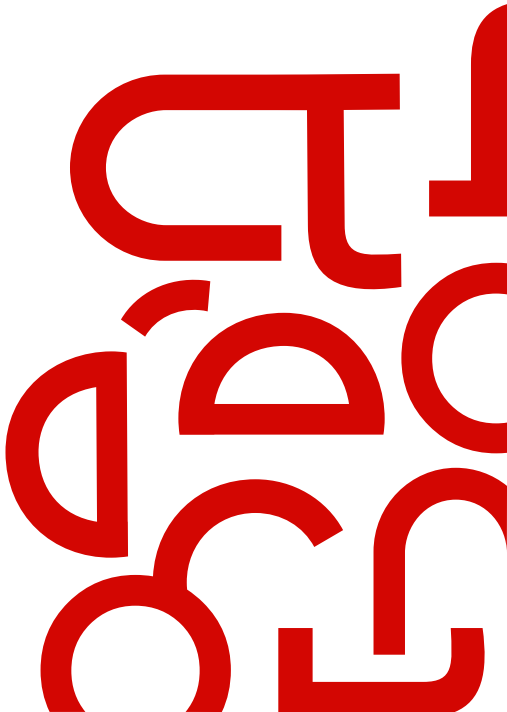
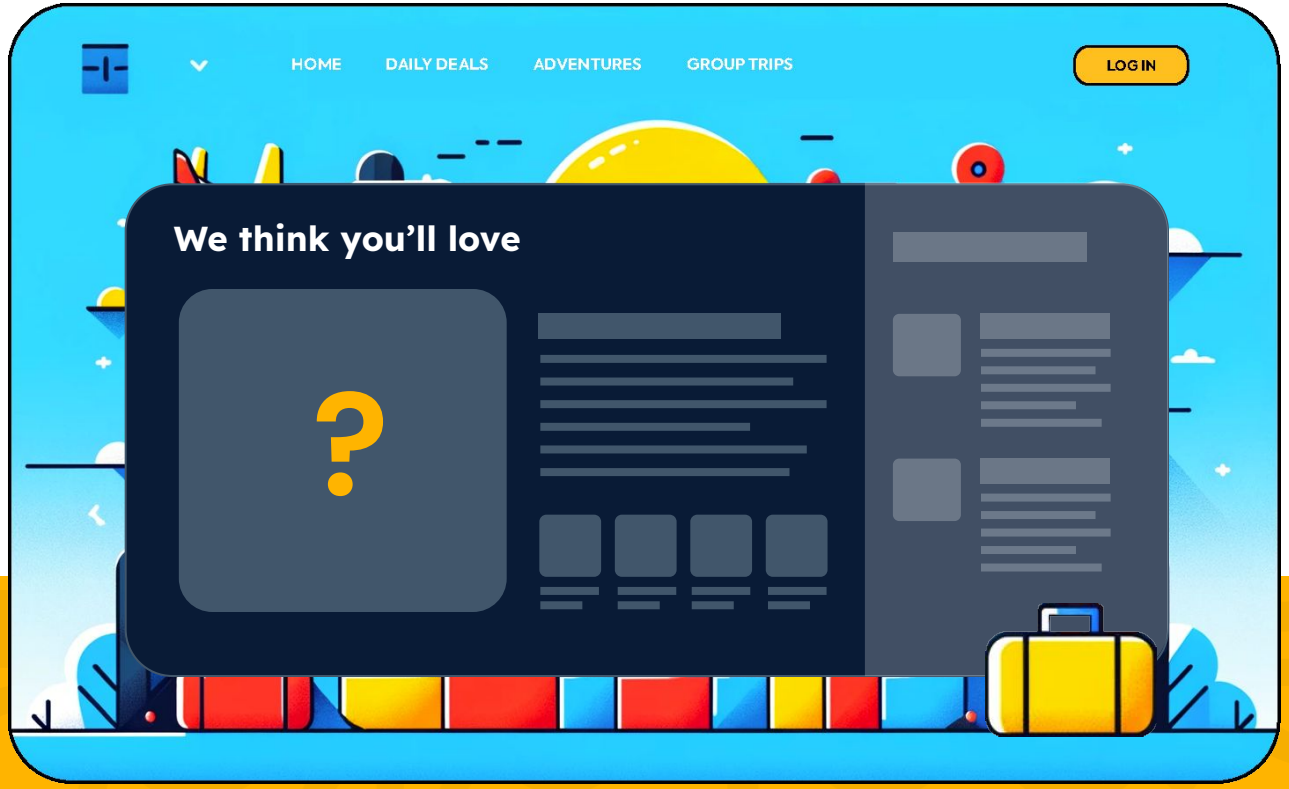
Full-RAG:
A modern architecture for hyper-personalization

Mike Del Balso
CEO & Co-Founder

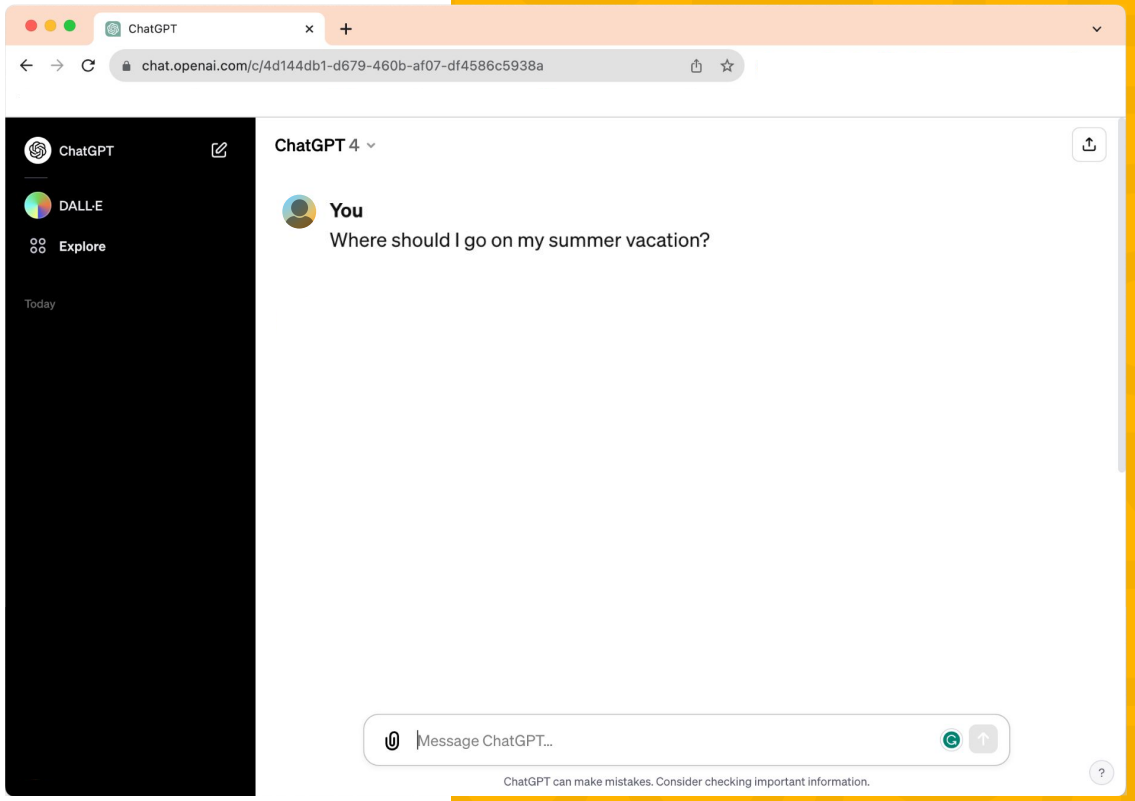
tecton



Goal: highly personalized travel recommendations

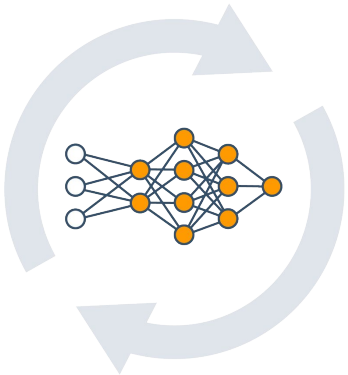


How can we get a good suggestion from a model?



How can we get a better recommendation?

Fine tune?

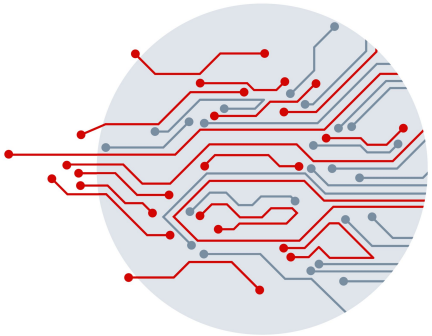


Improving model's intrinsic knowledge

Prompt Engineer?

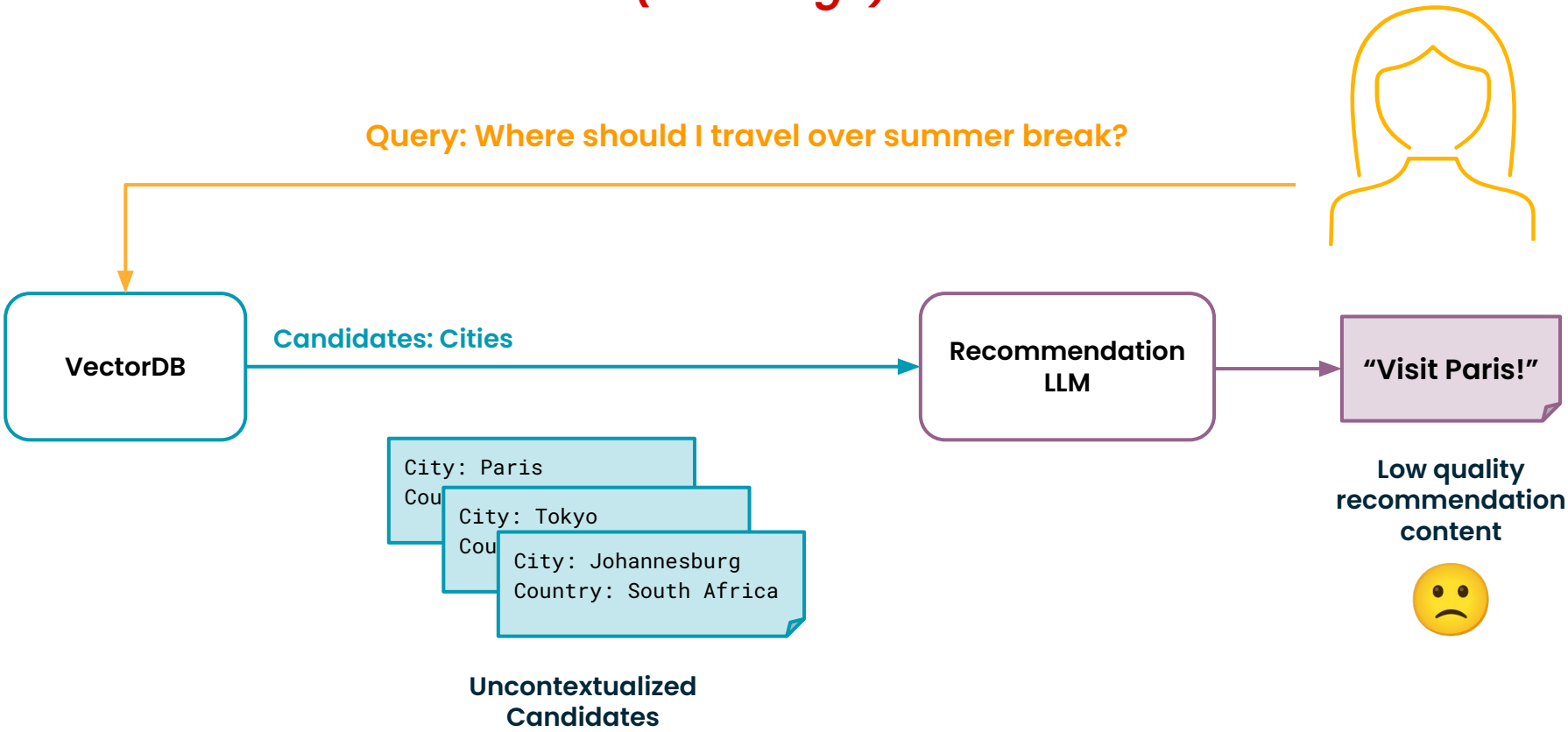
Rewording the question, giving time to think

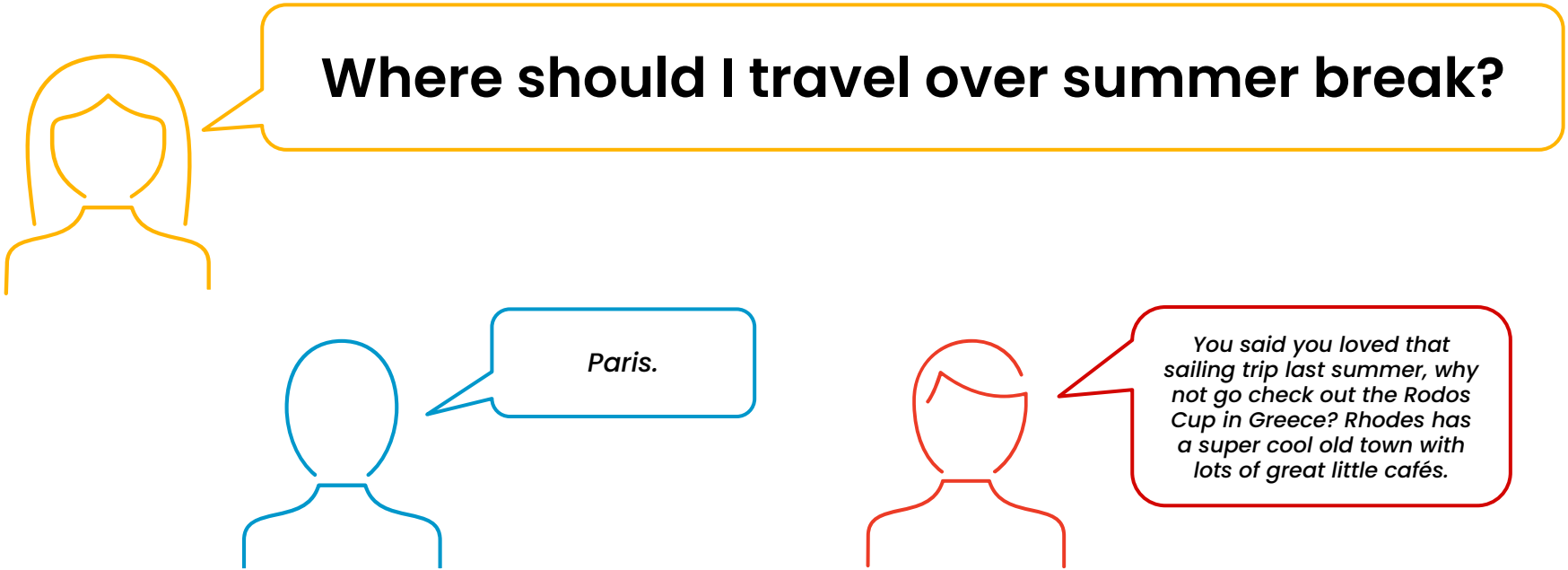
RAG?



Improving model's knowledge about the current situation

Traditional RAG
(Stone age)





Stranger Travel Agent
Low context, High expertise

Your Best Friend Travel Agent
High context, High Expertise

Without context (i.e. in Trad RAG), we just have uncontextualized candidates

Uncontextualized Candidate

```
City: Paris
Country: France
```

```
City: Tokyo
Country: Japan
```

Context is the **relevant information** that AI models use to understand a situation and make decisions.

Context enriches the candidate with more information to make it easier to reason about

Uncontextualized Candidate

```
City: Paris
Country: France
```

Contextualized Candidate

```
City: Paris
Country: France
Weather: 20°C, sunny
Activities: Museums, cafes, river tours
Nature: Fontainebleau, Versailles gardens
Events: Fashion Week, Bastille Day
Cuisine: Croissants, escargot
Language: French
Cost/Day: 200 USD
Safety: High
Visit Time: Apr-Jun, Sep-Nov
Accessibility: High, extensive public transport
Historic Sites: Eiffel Tower, Notre Dame
Accommodation Range: Hostels to luxury hotels
Visa Ease: Schengen Area, visa policies vary
Nightlife: Vibrant, diverse options
Family Friendly: Yes, many activities
Art Scene: Louvre, Montmartre
Shopping: Boutiques, flea markets
Internet Access: High-speed, widely available
```

Personalized Context enriches candidates with *user-level information*

Without context

```
City: Paris
Country: France
```

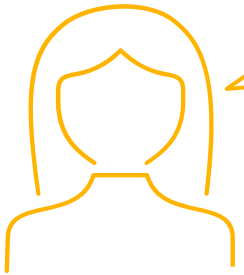
With context

```
City: Paris
Country: France
Weather: 20°C, sunny
Activities: Museums, cafes, river tours
Nature: Fontainebleau, Versailles gardens
Events: Fashion Week, Bastille Day
Cuisine: Croissants, escargot
Language: French
Cost/Day: 200 USD
Safety: High
Visit Time: Apr-Jun, Sep-Nov
Accessibility: High, extensive public transport
Historic Sites: Eiffel Tower, Notre Dame
Accommodation Range: Hostels to luxury hotels
Visa Ease: Schengen Area, visa policies vary
Nightlife: Vibrant, diverse options
Family Friendly: Yes, many activities
Art Scene: Louvre, Montmartre
Shopping: Boutiques, flea markets
Internet Access: High-speed, widely available
```

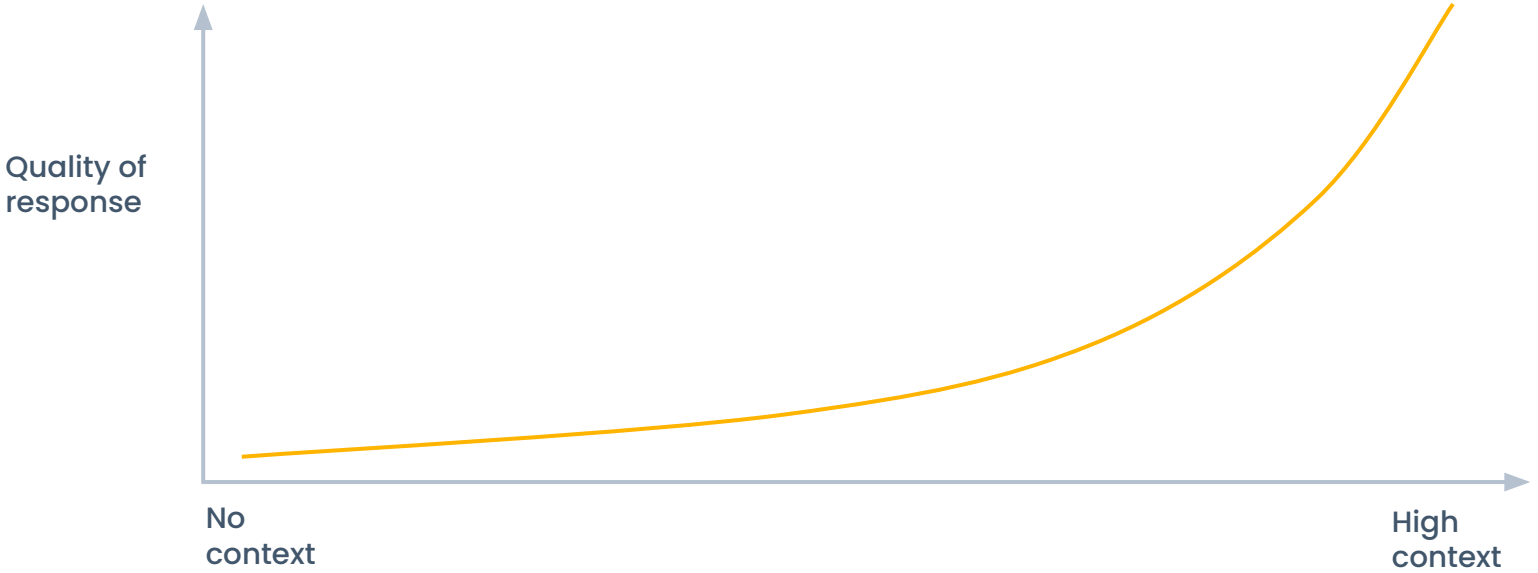
With Personalized Context

```
City: Paris
Country: France
Weather: 20°C, sunny
Activities: Museums, cafes, river tours
Nature: Fontainebleau, Versailles gardens
…
Preferred Climate: Mild
Interest in History: High
Dining Preference: Gourmet/Fine dining
Cultural Interest: High in arts and fashion
Budget: Luxury
Accommodation Preference: Boutique hotels
Preferred Language: Prefers English-friendly destinations
Activity Level: Moderate, enjoys leisurely strolls and seated
activities
Travel Experience: Seasoned traveler, prefers depth of experience
Travel Group: Solo traveler
Interest in Shopping: High, prefers unique boutiques
Nightlife Interest: Low, prefers quiet evenings
Interest in Local Cuisine: High, enjoys trying national dishes
Interest in Events: Moderate, selectively attends major events
Transportation Preference: Public transport, occasional taxi
```

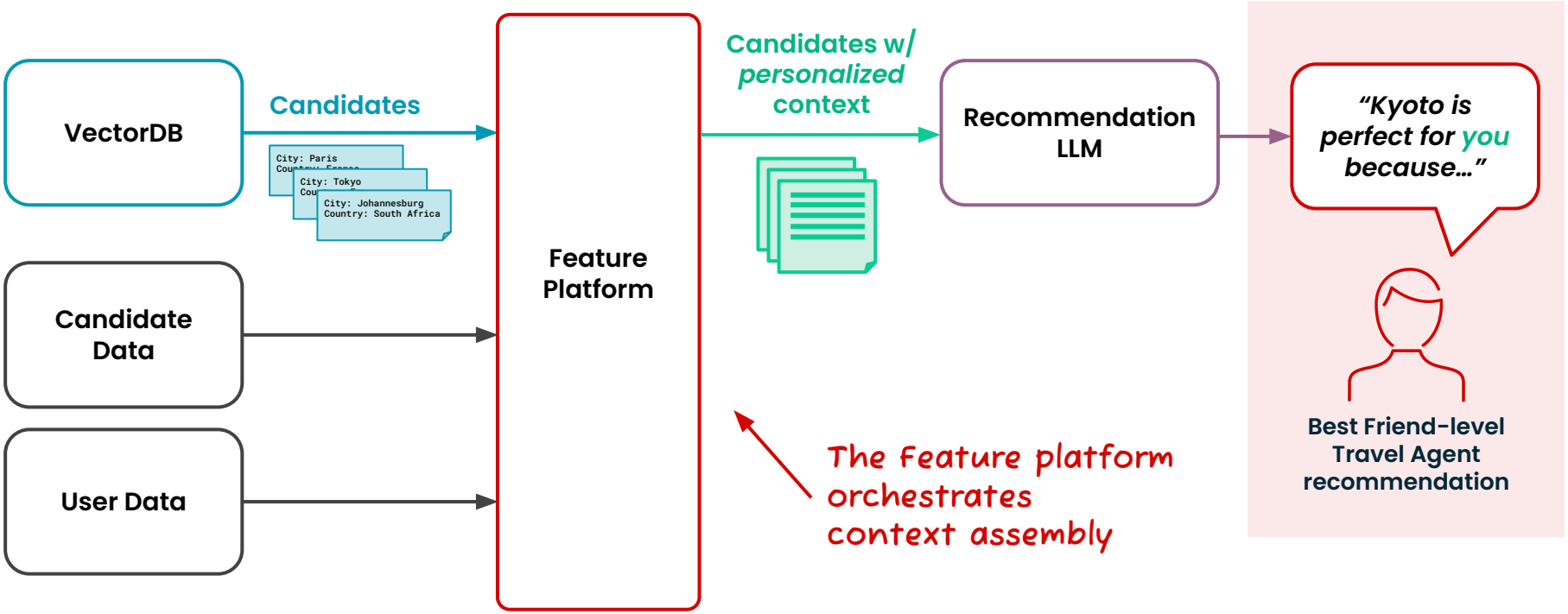
Examples of context



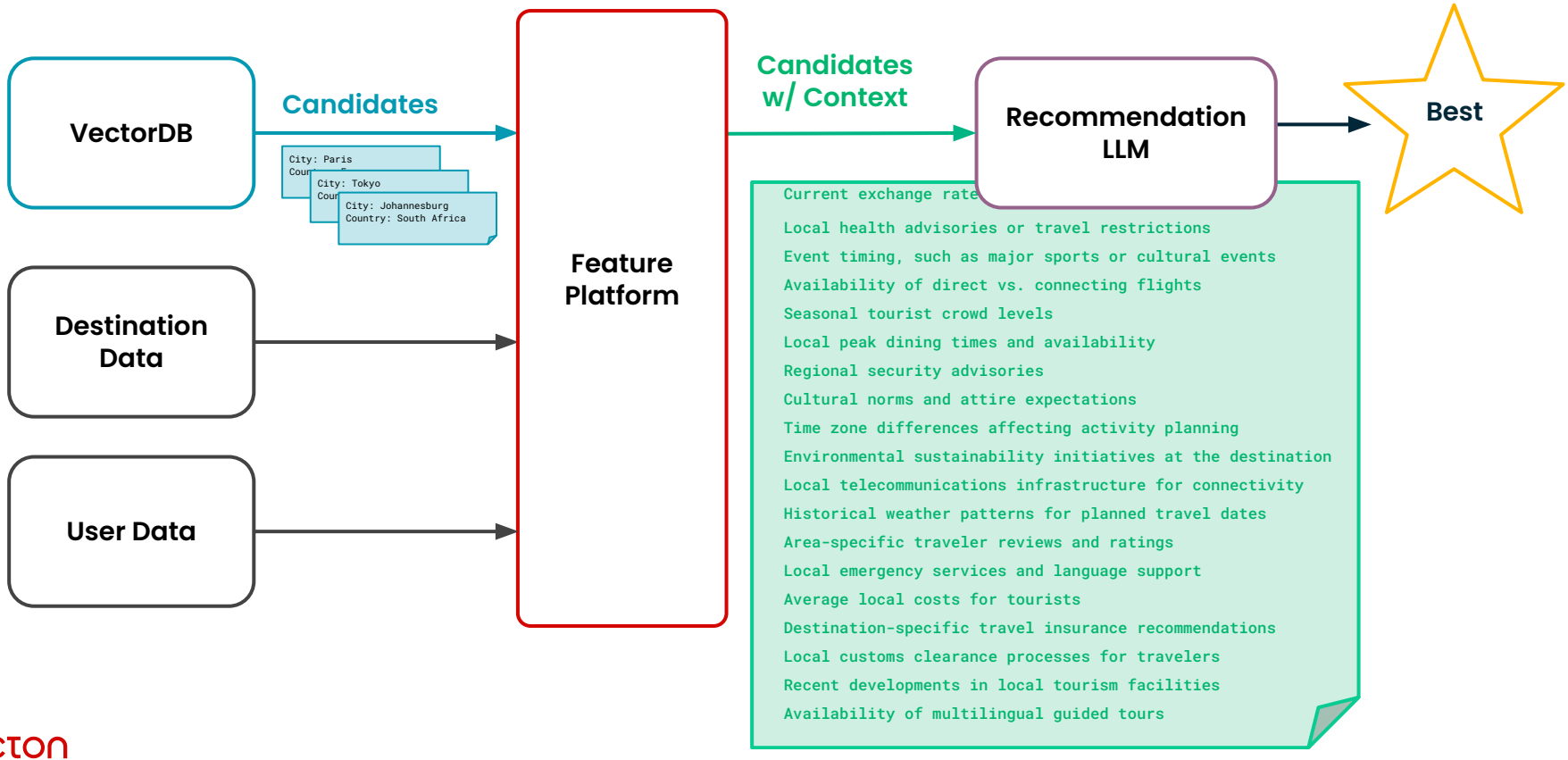
Where should I travel over summer break?

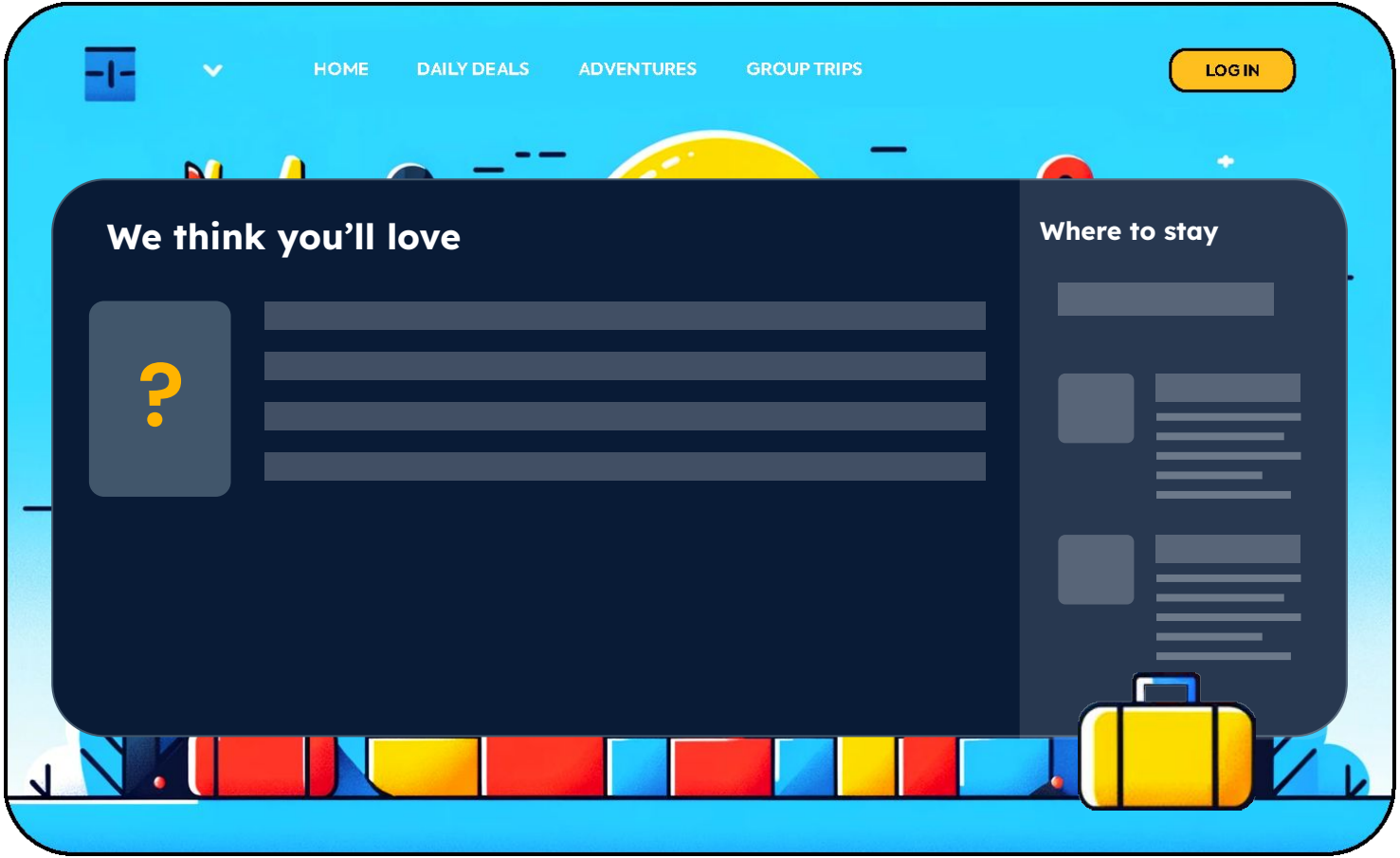


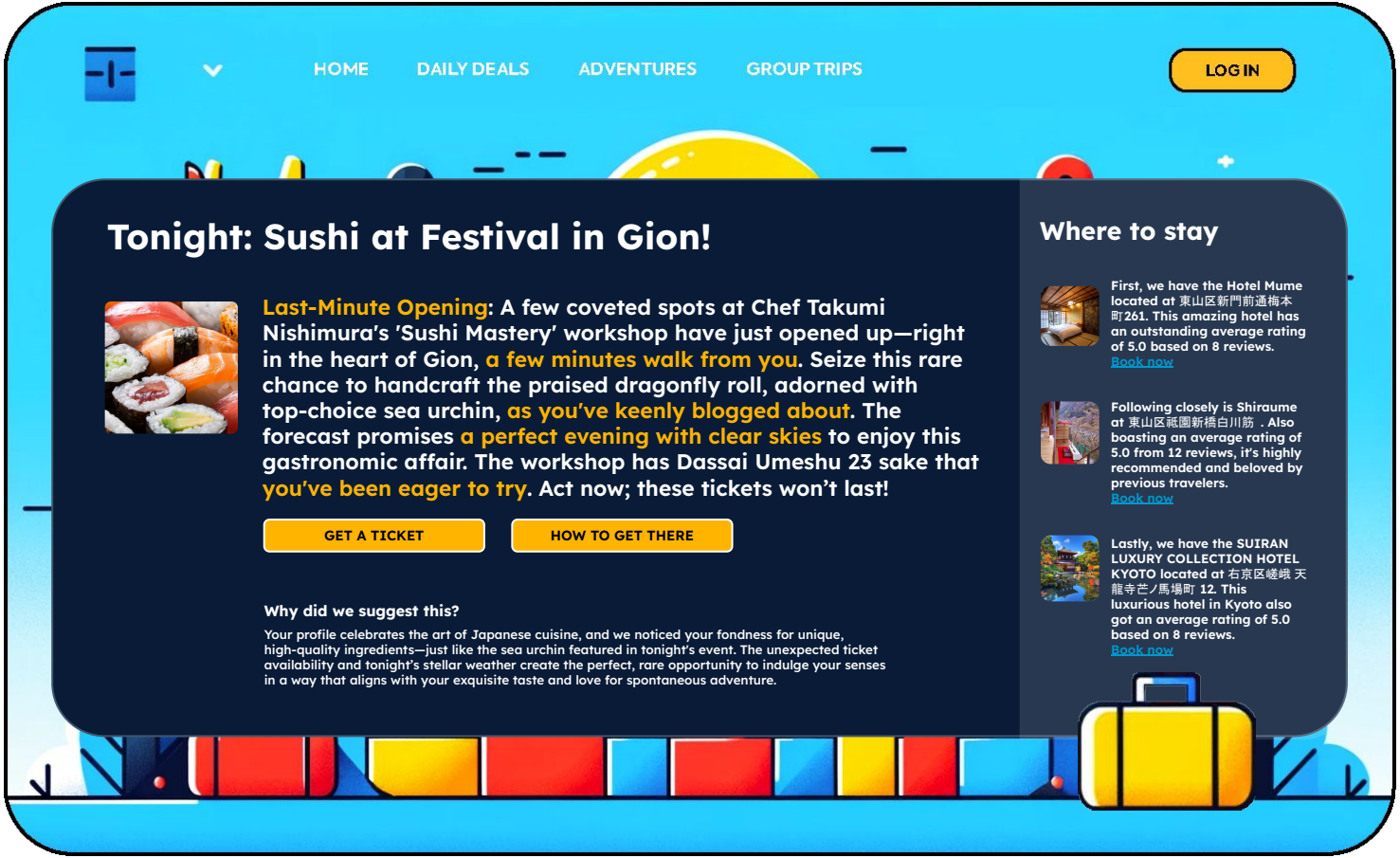
Create *personalized context* by enriching *candidates with relevant user data*



High personalization → Better recommendations




HOME
DAILY DEALS
ADVENTURES
GROUP TRIPS
LOG IN
We think you'll love
?
Where to stay


HOME
DAILY DEALS
ADVENTURES
GROUP TRIPS
LOG IN
Tonight: Sushi at Festival in Gion!
Last-Minute Opening: A few coveted spots at Chef Takumi Nishimura's 'Sushi Mastery' workshop have just opened up—right in the heart of Gion, a few minutes walk from you. Seize this rare chance to handcraft the praised dragonfly roll, adorned with top-choice sea urchin, as you've keenly blogged about. The forecast promises a perfect evening with clear skies to enjoy this gastronomic affair. The workshop has Dassai Umeshu 23 sake that you've been eager to try. Act now; these tickets won't last!
GET A TICKET
HOW TO GET THERE
Why did we suggest this?
Your profile celebrates the art of Japanese cuisine, and we noticed your fondness for unique, high-quality ingredients—just like the sea urchin featured in tonight's event. The unexpected ticket availability and tonight's stellar weather create the perfect, rare opportunity to indulge your senses in a way that aligns with your exquisite taste and love for spontaneous adventure.
Where to stay
First, we have the Hotel Mume located at 東山区新門前通梅本町261. This amazing hotel has an outstanding average rating of 5.0 based on 8 reviews.
Book now
Following closely is Shiraume at 東山区祇園新橋白川筋 . Also boasting an average rating of 5.0 from 12 reviews, it's highly recommended and beloved by previous travelers.
Book now
Lastly, we have the SUIRAN LUXURY COLLECTION HOTEL KYOTO located at 右京区嵯峨 天龍寺芒ノ馬場町 12. This luxurious hotel in Kyoto also got an average rating of 5.0 based on 8 reviews.
Book now

How can we build amazing personalized contexts?

4 Levels of context personalization

LEVEL 0:
No Context

LEVEL 3

LEVEL 2

LEVEL 1

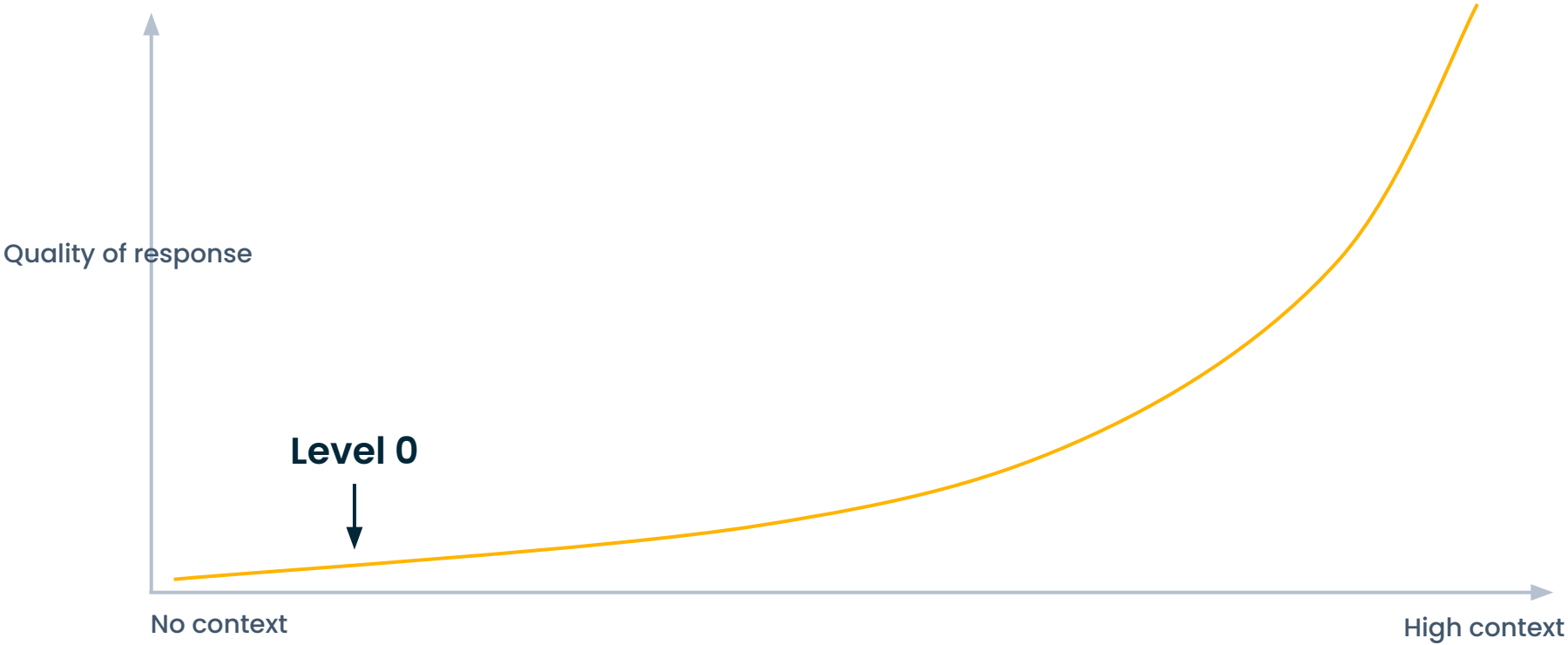
LEVEL 0

CONTEXT

- **None**

Broad, one-size-fits-all recommendations (TRAD RAG)

LEVEL 0:
No Context


Quality of response
Level 0
No context
High context

LEVEL 1:
Batch Context

LEVEL 3

LEVEL 2

LEVEL 1

CONTEXT
- **Batch**

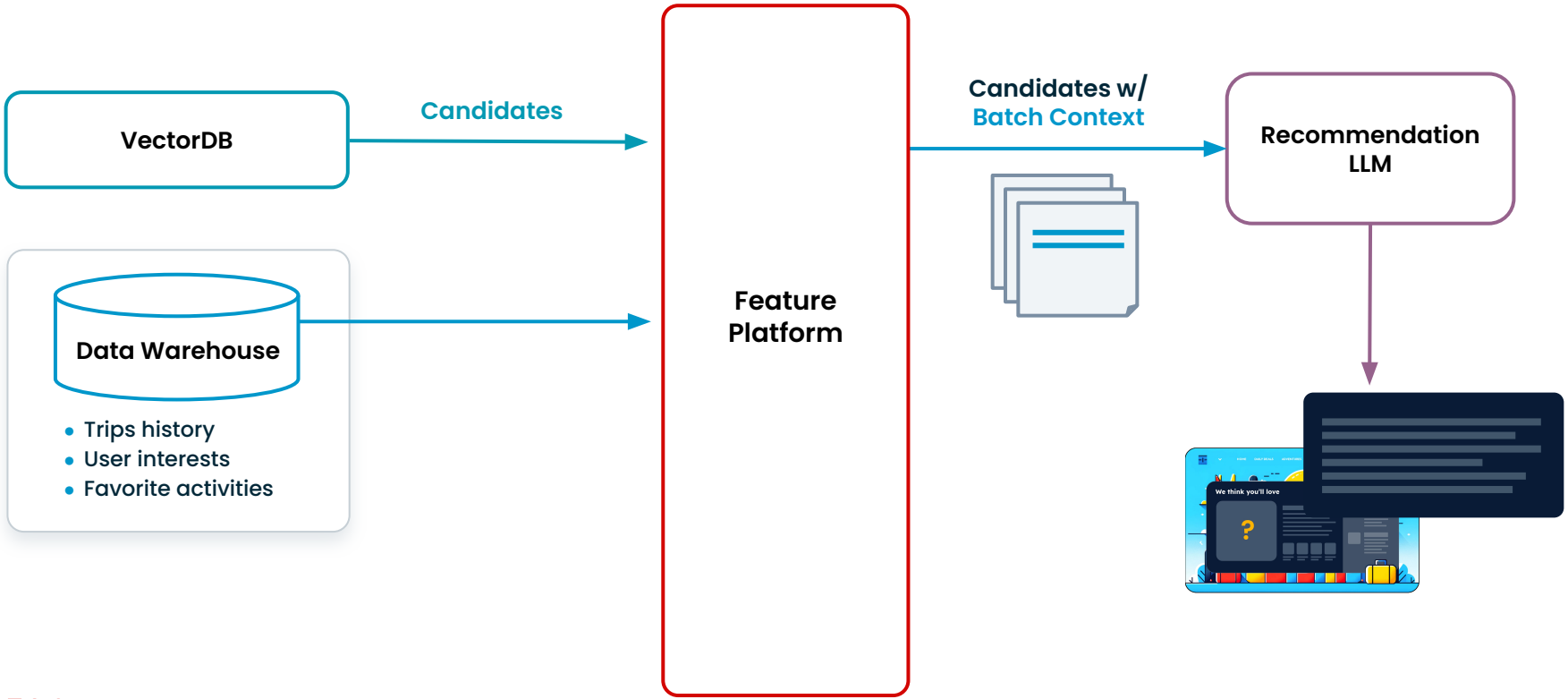
Personalized insights drawn from past behavior and profile data

LEVEL 0

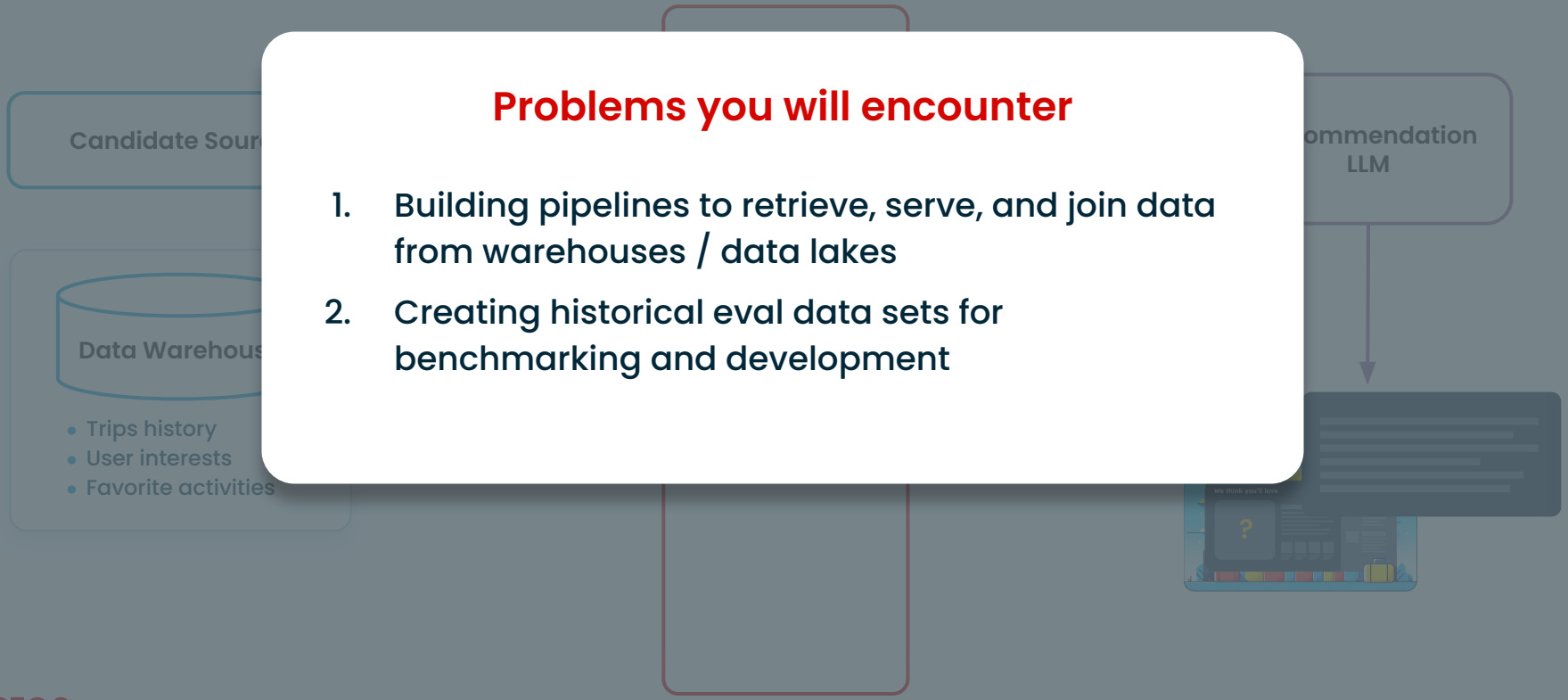
CONTEXT
- None

Broad, one-size-fits-all recommendations (The dumbest model)

LEVEL 1:
Batch Context



LEVEL 1:
Batch Context



Problems you will encounter

1. Building pipelines to retrieve, serve, and join data from warehouses / data lakes
2. Creating historical eval data sets for benchmarking and development

Building batch context simply

"What are the last 5 places this person has visited?"

1) Write simple definition

trip_history_features.py

```
@batch_feature_view(
    sources=[trips],
    entities=[user],
    mode='pandas',
    features=[Feature(column='destination', function=last(5))]
)
def users_last_5_trip_destinations(trips):
    return trips[['user_id', 'date', 'destination']]
```

2) Create Eval Data

```
df = users_last_5_trip_destinations.get_features_in_range("2022-01-01", "2023-11-14")
```

3) Deploy to production

```
$ tecton apply
```

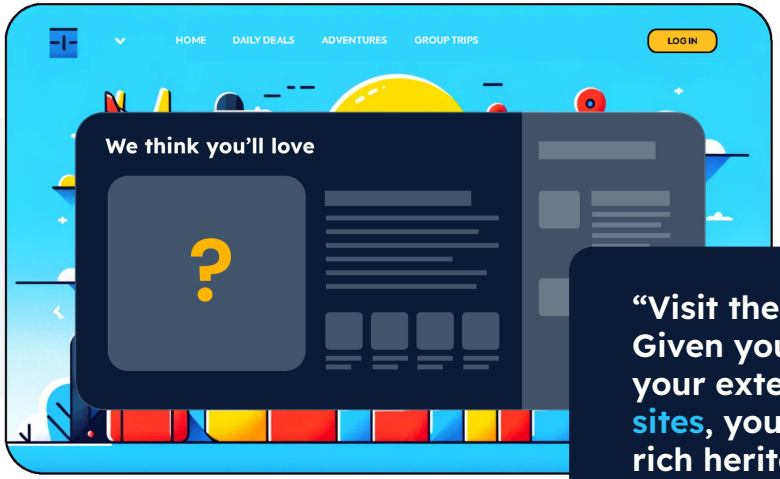
4) Read in real-time

```
context = tecton_client.get_features(feature_list)
```

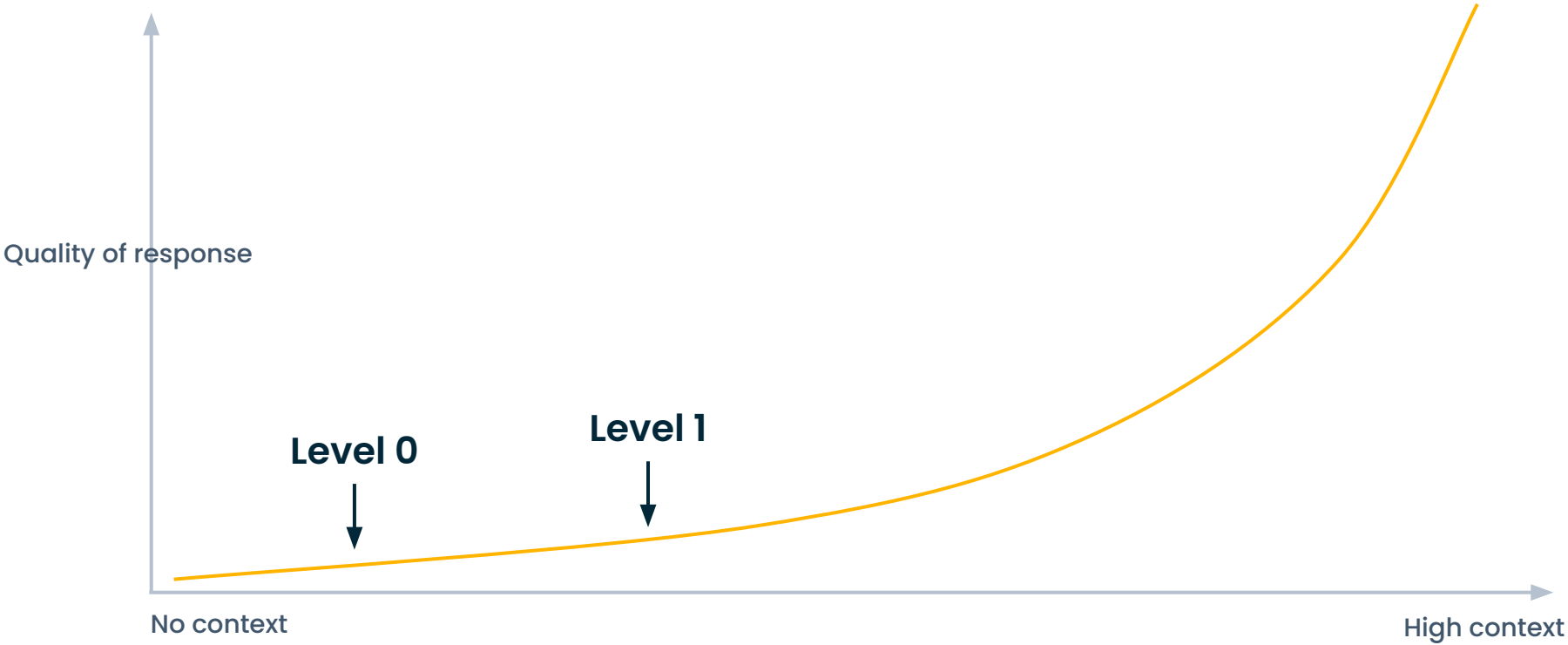
LEVEL 1:
Batch Context

Data Warehouse

- `trips_history`
- `user_interests`
- `favorite_activities`



"Visit the ancient city of Kyoto. Given your interest in history and your extensive travel to historical sites, you'll appreciate the city's rich heritage and numerous temples."

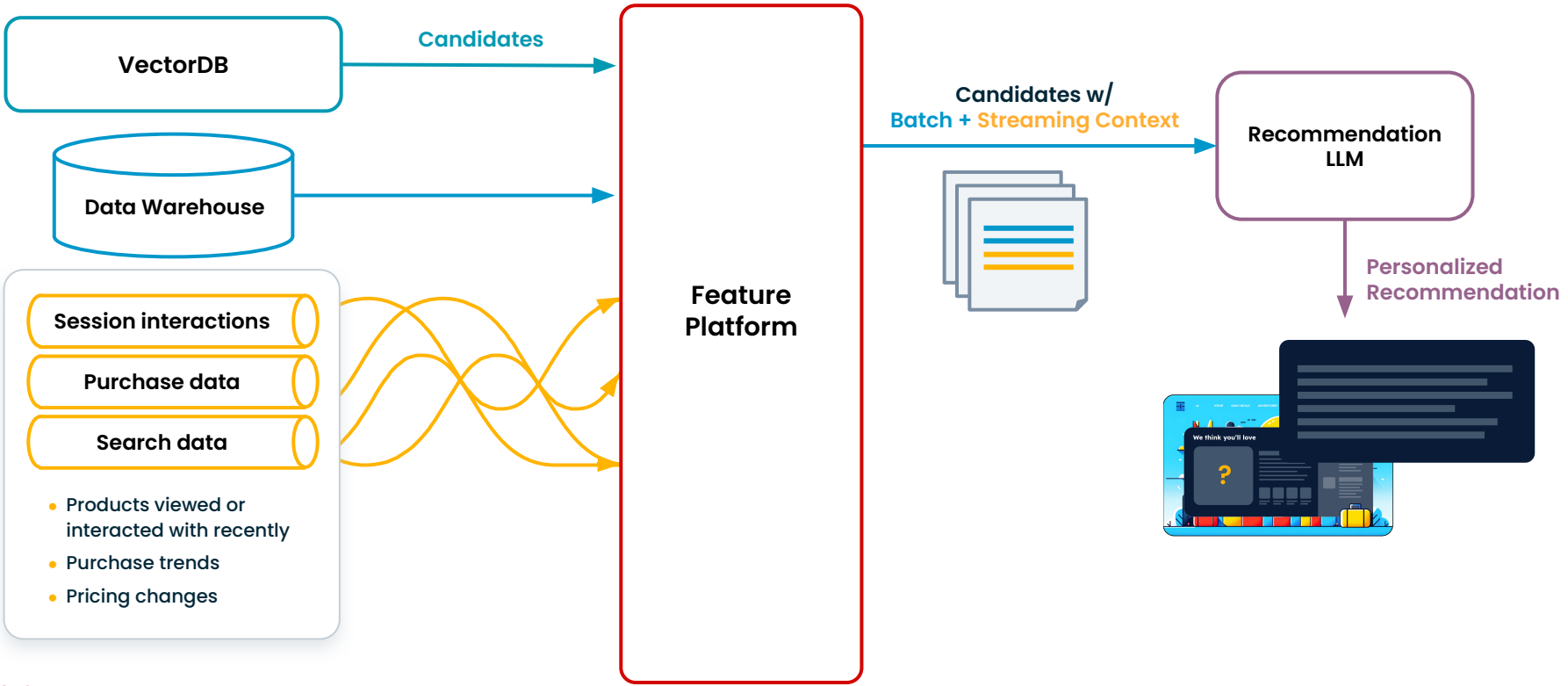

Quality of response
Level 0
Level 1
No context
High context

LEVEL 2:
Batch + Streaming Context

	CONTEXT	
LEVEL 2	• **Batch** • **Streaming**	Recommendations adapted to the user's current interests and interaction behavior
LEVEL 1	• Batch	Personalized insights drawn from past behavior and profile data
LEVEL 0	• None	Broad, one-size-fits-all recommendations

LEVEL 2:

Batch + Streaming Context



LEVEL 2:

Batch + Streaming Context

Problems you will encounter

1. Building, evaluating, productionizing, and monitoring streaming data pipelines 
2. Cost-efficient inference (not just the model!)

Building streaming context can also be simple

"In the past hour, what topics did the user watch a video about?"

1) Simple definition

media_interaction_features.py

```
@stream_feature_view(
    sources=[page_events],
    entities=[user],
    mode='pandas',
    features=[Feature(column='video_topic',
                      function=last_distinct(5),
                      time_window=timedelta(hours=1))]
)
def recent_video_interaction_topics(page_events):
   filtered_events = page_events[page_events['type'] = 'video_play']
   return filtered_events[['user_id', 'date', 'video_topic']]
```

2) Create Eval Data

```
df = users_last_5_trip_destinations.get_features_in_range("2022-01-01", "2023-11-14")
```

3) Deploy to production

```
$ tecton apply
```

4) Read in real-time

```
context = tecton_client.get_features(feature_list)
```

Building streaming context can also be simple

"In the past hour, what topics did the user watch a video about?"

1) Simple definition

media_interaction_features.py

```
@stream_feature_view(
    sources=[page_events],
    entities=[user],
    mode='pandas',
    features=[Feature(column='video_topic',
                      function=last_distinct(5),
                      time_window=timedelta(hours=1))]
)
def recent_video_interaction_topics(page_events):
   filtered_events = page_events[page_events['type'] = 'video_play']
   return filtered_events[['user_id', 'date', 'video_topic']]
```

2) Create Eval Data

```
df = users_last_5_trip_destinations.get_features_in_range("2022-01-01", "2023-11-14")
```

3) Deploy to production

```
$ tecton apply
```

Same workflow for any context

4) Read in real-time

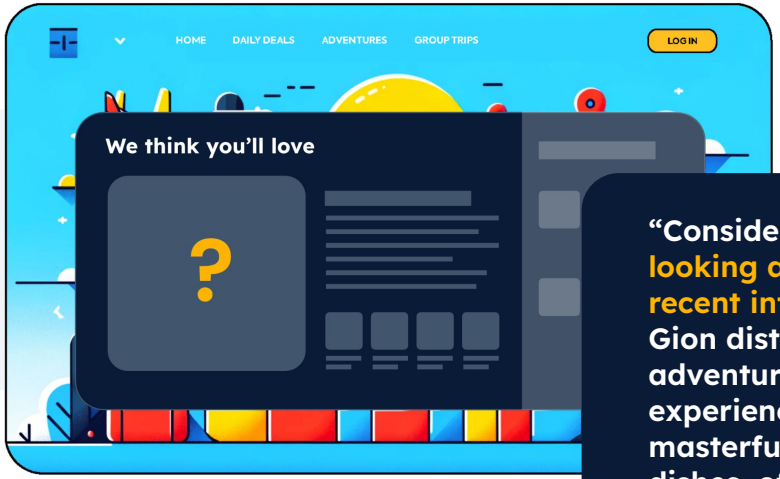
```
context = tecton_client.get_features(feature_list)
```

tecton

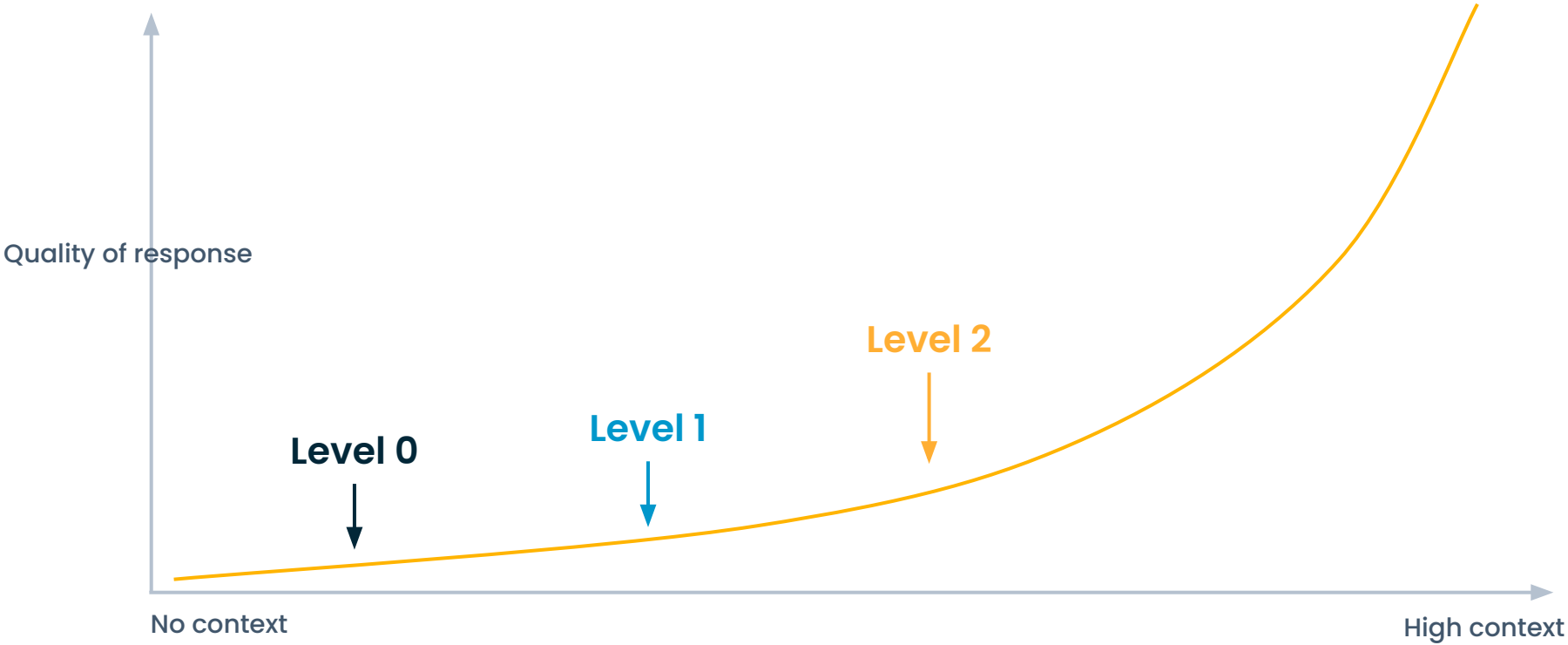
LEVEL 2:
Batch + Streaming Context

Streaming

- `locations_viewed_recently`
- `recent_activities_viewed`
- `pricing_changes`



"Considering you've recently been looking at trips to Japan and your recent interest in fine dining, Kyoto's Gion district presents a unique dining adventure with its renowned kaiseki experience. Seasonal ingredients are masterfully crafted into exquisite dishes, offering a feast for the senses. Don't miss this chance to indulge in Japan's artful cuisine during your stay!"


Quality of response
Level 0
Level 1
Level 2
No context
High context

LEVEL 3:
Batch + Streaming + Real-time Context

LEVEL 3

CONTEXT
- **Batch**
- **Streaming**
- **Real-time**

Informed, personalized recommendations using live external events, the user's current context, and real-time inputs

LEVEL 2

CONTEXT
- Batch
- Streaming

Recommendations adapted to the user's current interests and interactive behavior

LEVEL 1

CONTEXT
- Batch

Personalized insights drawn from past behavior and profile data

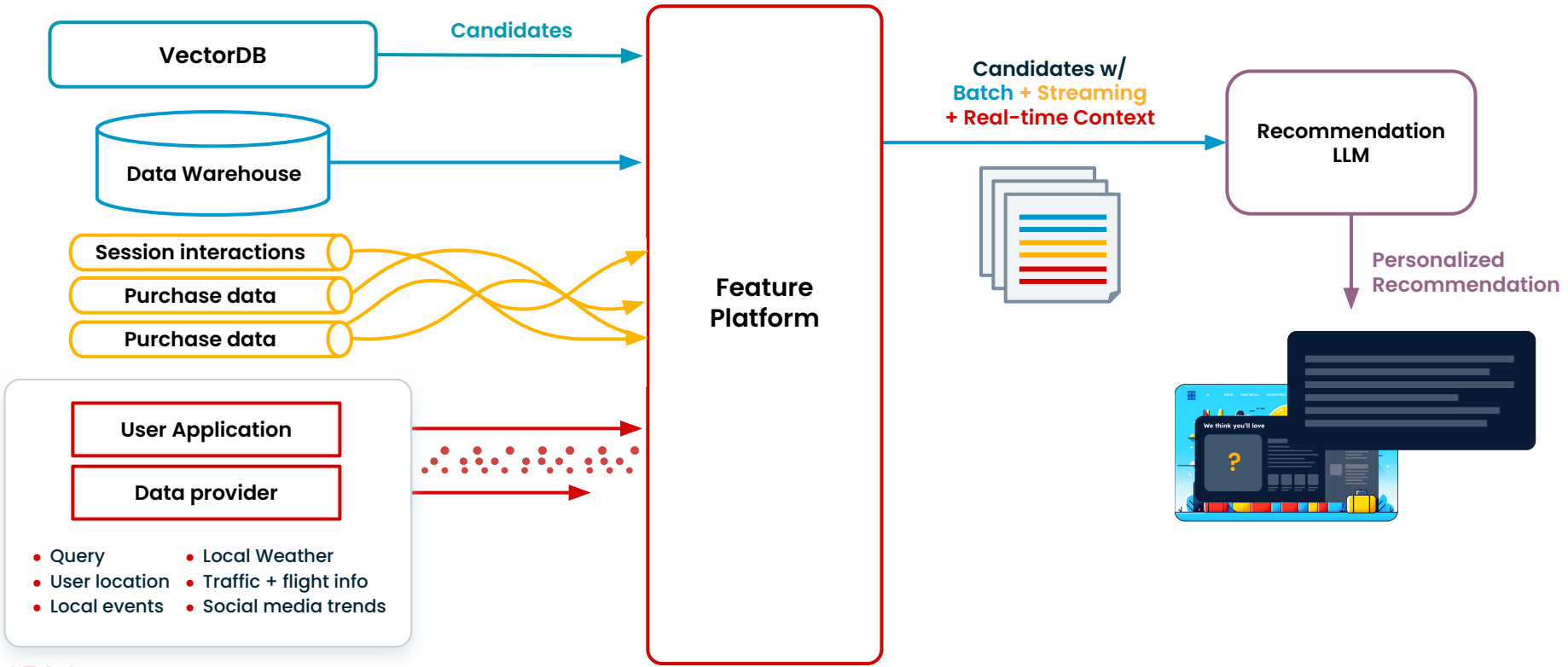
LEVEL 0

CONTEXT
- None

Broad, one-size-fits-all recommendations

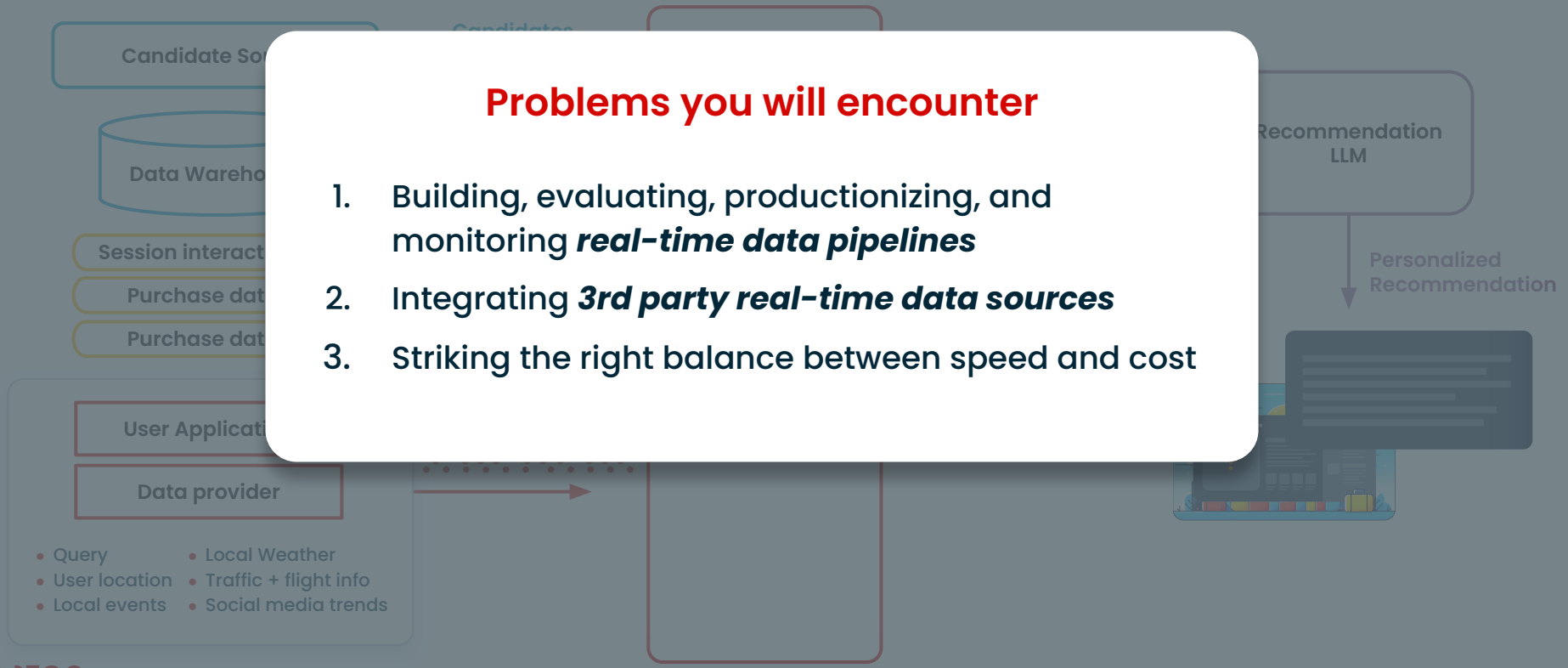
LEVEL 3: Full RAG

Batch + Streaming + Real-time Context



LEVEL 3:

Batch + Streaming + Real-time Context



Problems you will encounter

1. Building, evaluating, productionizing, and monitoring ***real-time data pipelines***
2. Integrating ***3rd party real-time data sources***
3. Striking the right balance between speed and cost

Building real-time context works the same way

"How far is the user from the destination? Same country?"

1) Write simple definition

device_destination_distance_features.py

```python
@on_demand_feature_view(
    sources=[device_info, dest_info],
    mode='python',
    features=[
        Feature('distance_to_destination'),
        Feature('same_country')
    ]
)
def device_to_destination_distance_features(device_info, dest_info):
    result = {}

    device_location = (device_info['latitude'], device_info['longitude'])
    dest_location = (dest_info['lat'], dest_info['long'])

    result['distance_to_destination'] =
        geodesic(device_location, dest_location).kilometers

    result['same_country'] = device_info['country_id'] == dest_info['country_id']
    return result
```

...the other steps are the same

tecton

Building real-time context works the same way

"What's the weather like in that place *right now*?"

1) Write simple definition

destination_weather_features.py

```python
@api_source(
    sources=[destination_info],
    cache_ttl = timedelta(minutes=30)
)
def weather_data(destination_info):
    lat = destination_info['latitude']
    long = destination_info['longitude']

    api_key = tecton.secrets.get('WEATHER_API_KEY')
    url = f'https://api.weatherapi.com/v1/current.json?key={api_key}&q={lat},{long}'

    return requests.get(url).json

@on_demand_feature_view(
    sources=[weather_data],
    mode='python',
    features=[
        Feature('current_temperature', Float64),
        Feature('weather_condition', String)
    ]
)
def weather_features(weather_data):
    features = {}

    features['current_temperature'] = weather_data['current']['temp_c']
    features['weather_condition'] = weather_data['current']['condition']['text']

    return features
```

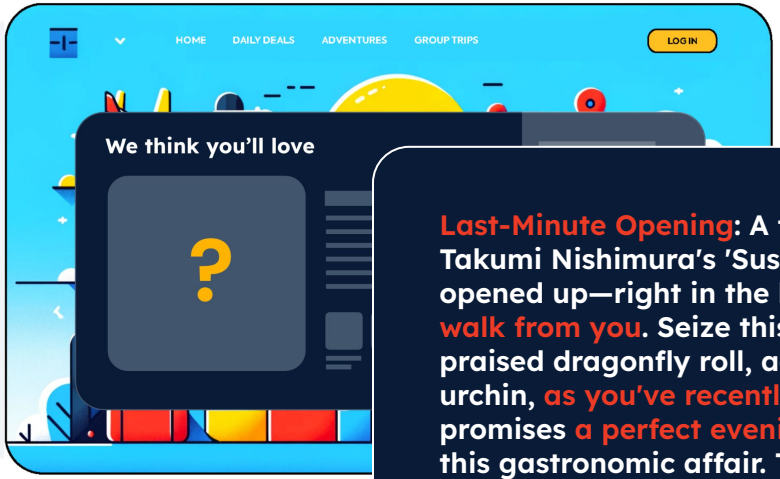
...the other steps are the same

tecton

LEVEL 3:
Batch + Streaming + Real-time Context

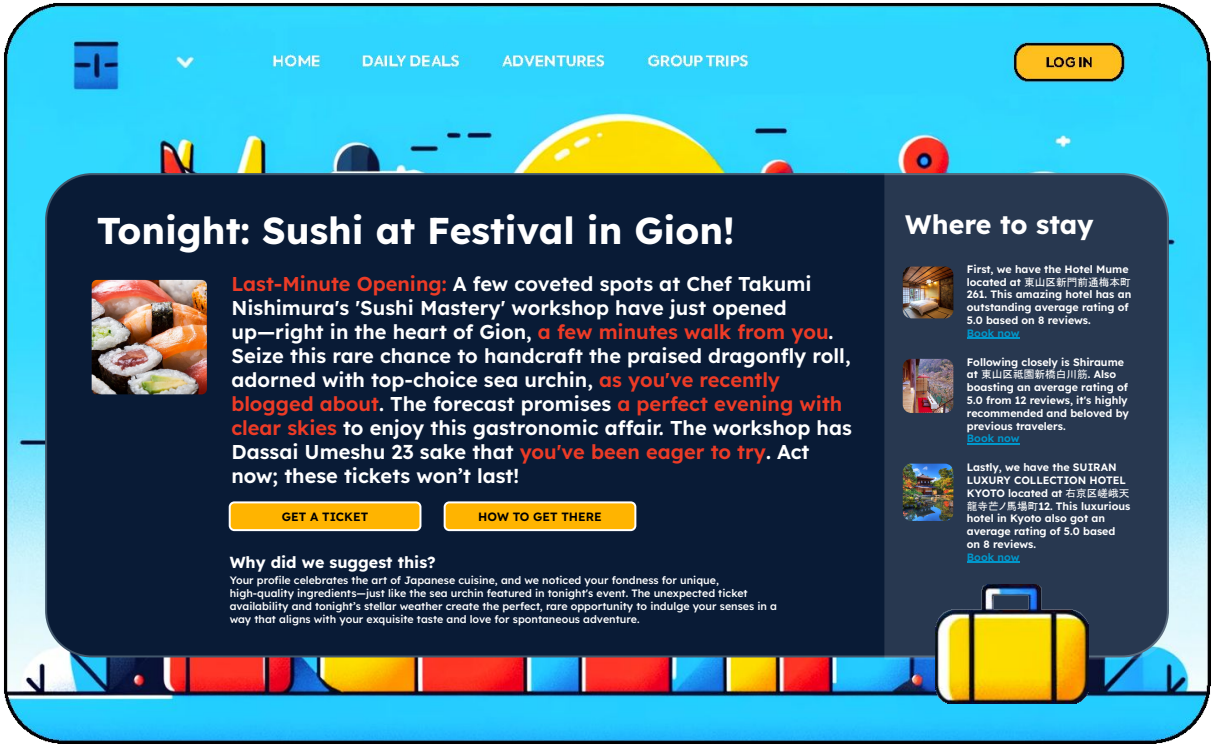
Real-time

- `query`
- `user_location`
- `local_events`
- `local_weather`
- `traffic_and_flights`
- `social_media_trends`



Last-Minute Opening: A few coveted spots at Chef Takumi Nishimura's 'Sushi Mastery' workshop have just opened up—right in the heart of Gion, a few minutes walk from you. Seize this rare chance to handcraft the praised dragonfly roll, adorned with top-choice sea urchin, as you've recently blogged about. The forecast promises a perfect evening with clear skies to enjoy this gastronomic affair. The workshop has Dassai Umeshu 23 sake that you've been eager to try. Act now; these tickets won't last!

Real-time personalization means more trusted and valuable recommendations



BONUS LEVEL 4

Real-time Context w/ feedback

LEVEL 4 IN CONCEPT	CONTEXT	• **Batch** • **Streaming** • **Real-time** **with feedback**	Informed, personalized recommendations using live external events, the user's current context, and real-time inputs
LEVEL 3	CONTEXT	• Batch • Streaming • Real-time	Informed, personalized recommendations using live external events, the user's current context, and real-time inputs
LEVEL 2	CONTEXT	• Batch • Streaming	Recommendations adapted to the user's current interests and interactive behavior
LEVEL 1	CONTEXT	• Batch	Personalized insights drawn from past behavior and profile data
LEVEL 0	CONTEXT	• None	Broad, one-size-fits-all recommendations

OK, what did we learn?

Context is King!

E-commerce	*Tailored shopping experiences*
Communication	*Conversational AI that understands you*
Content	*Recommendations that resonate*
Health & Wellness	*Customized wellbeing plans*
Financial Services	*Personal financial advice*

1

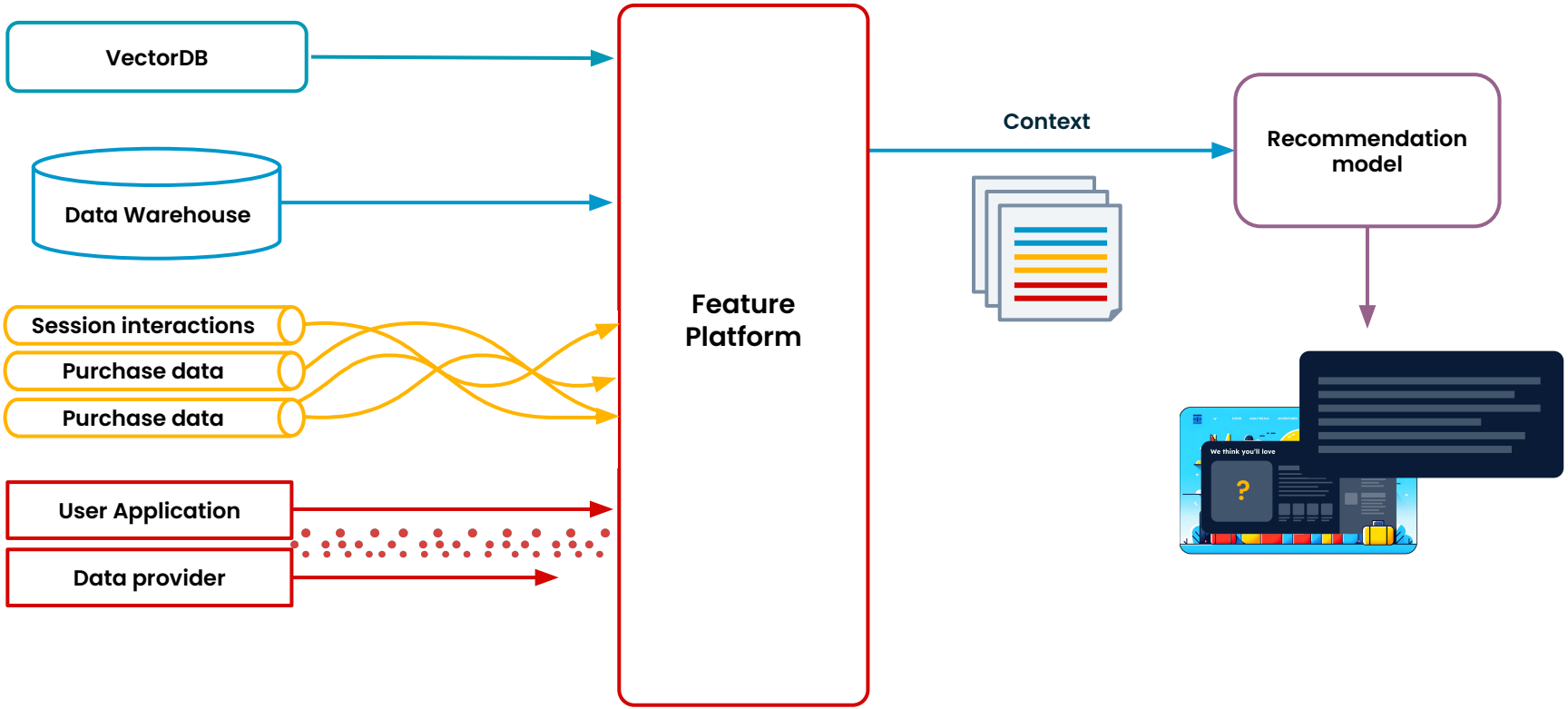
***Personalizing* context** can unlock amazing AI behaviors and product experiences.

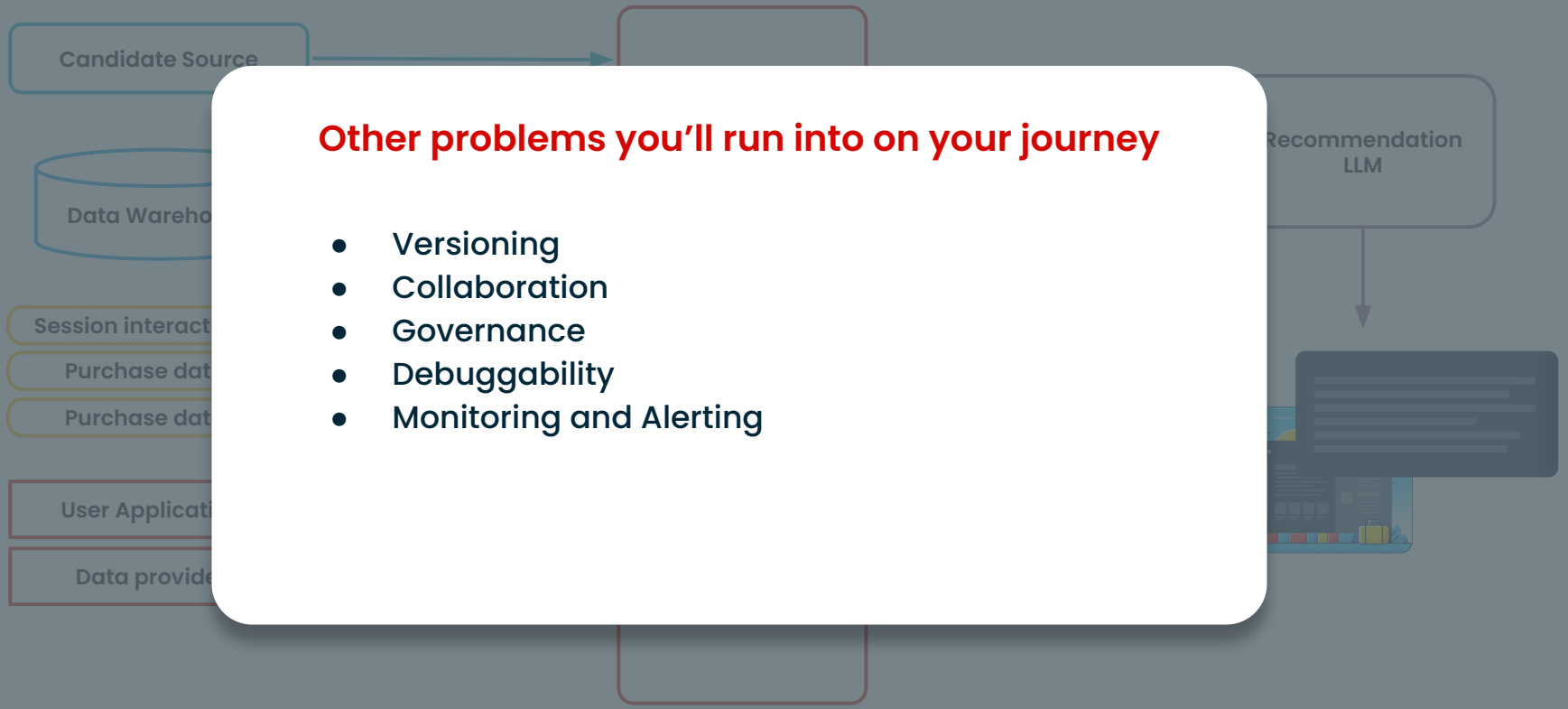
2

Higher degrees of personalization are **more valuable but harder to build**.

3

Feature Platforms can configure and assemble personalized context for LLMs.


VectorDB
Data Warehouse
Session interactions
Purchase data
Purchase data
User Application
Data provider
Feature Platform
Context
Recommendation model
We think you'll love
?

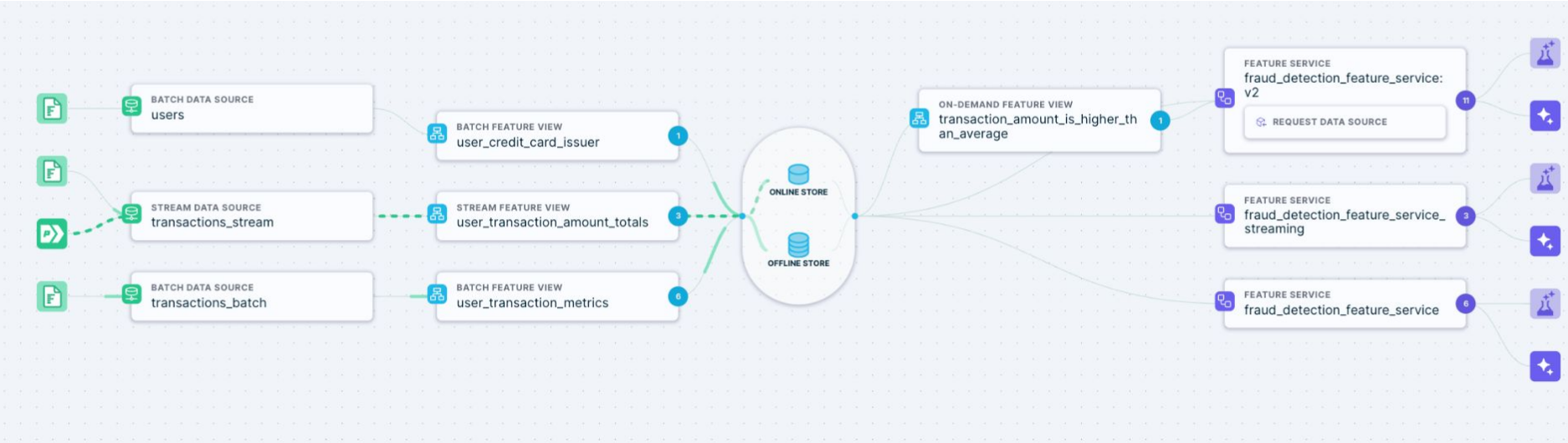


Other problems you'll run into on your journey

- Versioning
- Collaboration
- Governance
- Debuggability
- Monitoring and Alerting

Build a Full RAG today

...and solve all your other AI data problems



Get started at *tecton.ai/explore*

ANNOUNCING

Rift is now in Public Preview

The world's fastest path to real-time AI.

Python is all you need
Python transformations
for batch, streaming, & real-time.

Lightning-fast iteration
Develop & test locally.
Productionize instantly.

Unmatched performance
Millisecond-fresh aggregations
across millions of events.

Try Rift now: **tecton.ai/explore**

tecton