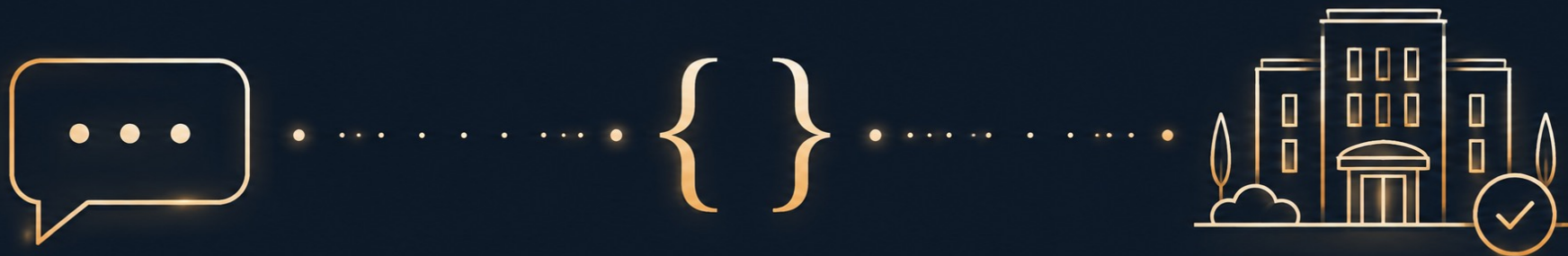


AI-Powered Frontend:

Integrating LLMs Into Your Angular App Without the Hype



Who has built an AI demo?

And who has had to
support it afterward?

That's the part I want to talk about.

A LITTLE ABOUT ME



Scan to visit sayeed.in

Hi, I'm Sayeed

Principal Frontend Architect at Ascendion

MY BACKGROUND

Financial services and healthcare
Airlines and government

WHAT I'M WORKING ON

Bringing LLMs into frontend apps
that teams can understand and support.

THE IDEA

What happens after someone types a request?



The model proposes the filters.
Our application decides what to apply.

THE REQUEST

“Show hotels in Dallas under \$200/night with a pool.”

CITY Dallas

CAP Less than \$200 per night

NEEDS Pool

My filter needs fields it can check.

ONE RESPONSE

The chat and the results belong together.

ONE CLAUDE RESPONSE

TEXT

“I’ll search Dallas...”

↓ Ivy’s chat

TOOL CALL

`filter_hotels`

city + price cap + amenities

↓ Validate → filter → cards

HOW THE PIECES CONNECT

The Angular service calls our backend.



POST /api/chat



Express / Node.js

Prompt + schema



Angular service



Our backend



Claude

Interaction and UI state
Local hotel filtering

API key, prompt,
validation, timeout

Text and
proposed
arguments

← Streamed text + validated criteria

← Model response

I keep the API key out of the browser.

YOUR CALL

Would you apply this response yet?

```
{"locationQuery":"Dallas","maxPrice":2
```

- A** Start searching
- B** Wait for the complete, validated call

Meet Ivy, Lodgic's hotel assistant.

Lodgic = Lodge + Logic

“Dallas, under \$200, with a pool.”

- 01 Watch Ivy's reply arrive
- 02 Inspect the staged and committed filters
- 03 Watch the hotel cards change

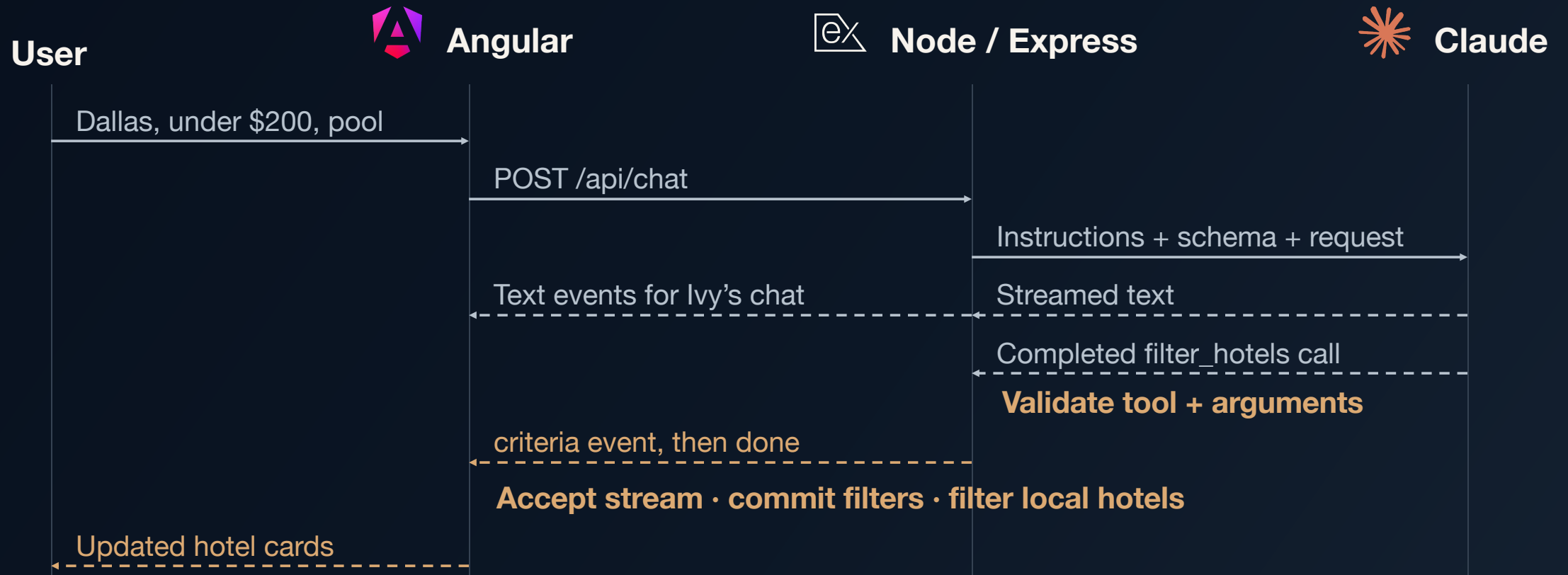
Let's find a hotel



“Show hotels in Dallas under \$200/night with a pool.”

Watch Ivy's reply stream in,
then watch the hotel results update.

What happened behind that search?



Claude proposes the filters. Our code validates them and searches the hotel data.

What I send to Claude

PROMPT RULES

Use null if no price cap was requested.
Describe the search you will run.
Do not claim that results have been found.

`filter_hotels` · example accepted arguments

```
{ "locationQuery": "Dallas", "maxPrice": 200,  
  "amenities": ["pool"], "minRating": null,  
  "selectedHotelId": null }
```

Here's what I'll let the app accept.

```
const Criteria = z.object({  
  locationQuery: z.string().trim().min(1).max(100),  
  maxPrice: z.number().positive().max(10000).nullable(),  
  amenities: z.array(Amenity).max(12),  
  minRating: z.number().min(0).max(5).nullable(),  
  selectedHotelId: z.string().min(1).nullable(),  
}).strict();
```

The prompt asks. Application code checks.

Text arrives first. Filters pass a gate.

```
stream.on('text', (delta) => {
  write({ type: 'text', requestId, delta });
});
const final = await stream.finalMessage();
const result = acceptFinalMessage(final);
if (result.kind === 'search') {
  write({ type: 'criteria', requestId,
    toolId: result.toolId, criteria: result.criteria });
  write({ type: 'done', requestId, outcome: 'search' });
}
```

Ivy can speak while Lodgic waits for valid filters.

Stage the filters. Commit after success.

WHEN THE CRITERIA EVENT ARRIVES

```
this.stagedCriteria.set(criteria);
```

AFTER `send()` RESOLVES SUCCESSFULLY

```
if (!owns()) return;  
if (outcome === 'search') {  
  this.activeCriteria.set(this.stagedCriteria());  
  this.selectedHotelId.set(null);  
}
```

filteredHotels recomputes. Angular updates the cards.

What should we break first?

- A** The response stops halfway through
- B** The model sends an invalid price

Then we'll recover and run the search again.

What does “done” mean?

- 01 Streaming: text is still arriving
- 02 Validating: should we accept these filters?
- 03 Complete: accepted filters are applied

Cancelled, failed, and zero matches are different outcomes.

What happened to this request?

- 01 Request ID and model
- 02 Time to first text
- 03 Time to accepted filters
- 04 Validation outcome and failure stage

**Track when the reply starts, when filters arrive,
and where failures happen.**

What would I need at 2 a.m.?

- 01 A trace I can follow
- 02 A timeout and a retry budget
- 03 The prompt, schema, and model version
- 04 A clear failure state—and a way to recover

ONE THING TO TAKE BACK

Pick one small feature you own.



What would you let
a model decide?



What would your code
still have to check?

Take that question to your next design review.

QUESTIONS



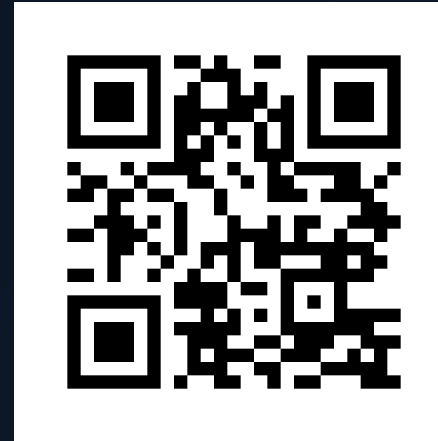
Let's talk.

Thanks for spending the time with me.



Let's connect

LinkedIn: /in/sayeed



Slides & demo code

sayeed.in/speaking