

The background features a dark blue gradient with large, overlapping, semi-transparent shapes in shades of purple and magenta. Two thin, light blue lines cross the scene diagonally. The text is positioned on the left side of the image.

AWS re:Invent

DECEMBER 2 - 6, 2024 | LAS VEGAS, NV

PRO301

The National Football League streamlines research and content creation

Eric Peters

(he/his/him)

Director, Post Production &
Media Administration
National Football League

Henry Wang

(he/his/him)

Sr. Applied Scientist
Amazon Web Services

Ross Claytor

Principle Applied Scientist
Amazon Web Services



Agenda

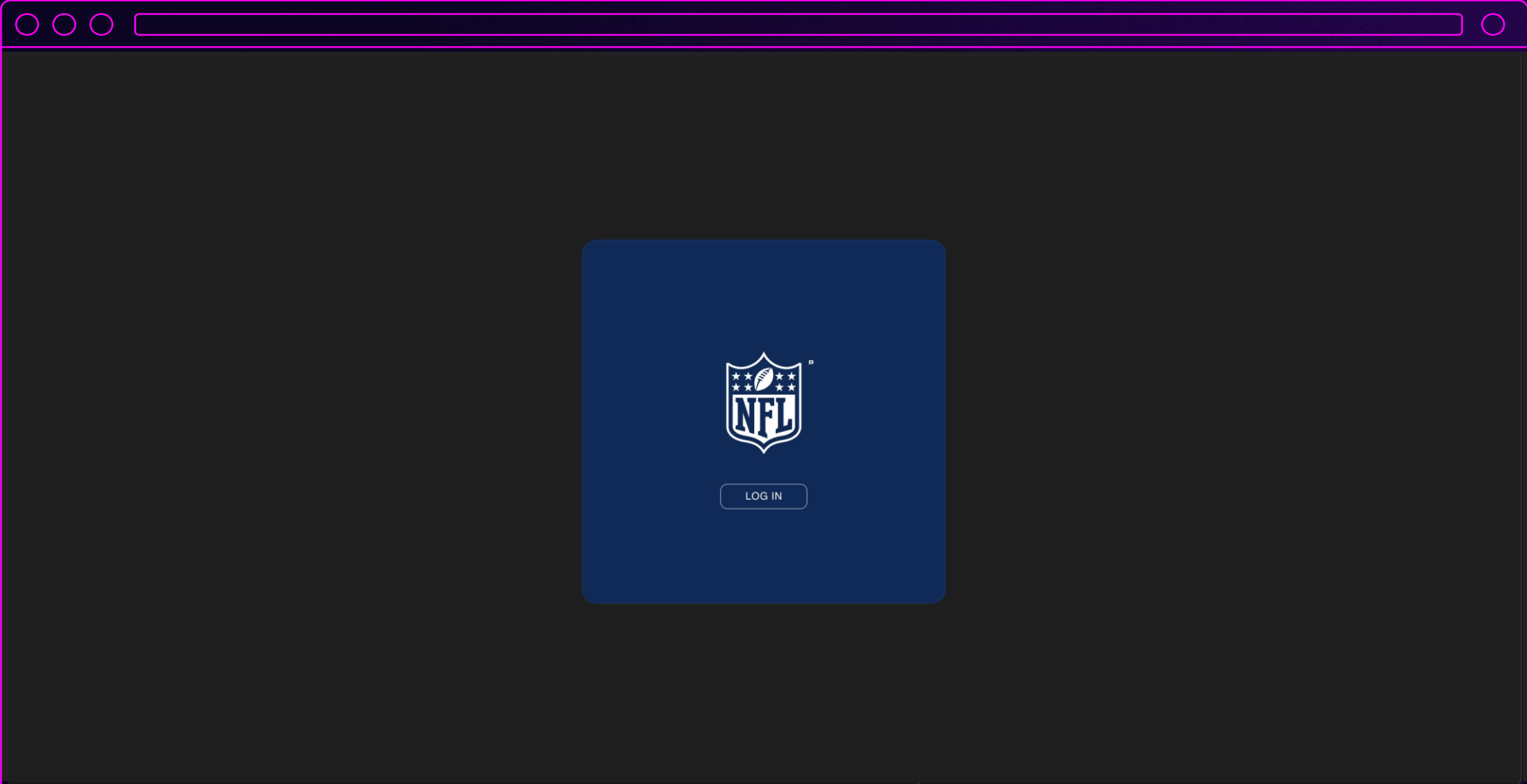
- 01 Content search with gen AI at NFL – Demo
- 02 Motivation for adopting gen AI
- 03 Building the MVP gen AI solution
- 04 Path to production
- 05 Impact and future work

NFL Media at a glance

- NFL Media is based in Los Angeles
- NFL Network was launched in 2003
- Media group has expanded over the years to include Red Zone, NFL.com, NFL+, NFL Channel, and more!
- Longstanding partnership with AWS including Next Gen Stats



Playbook Pro – A Next Gen Stats-based LLM



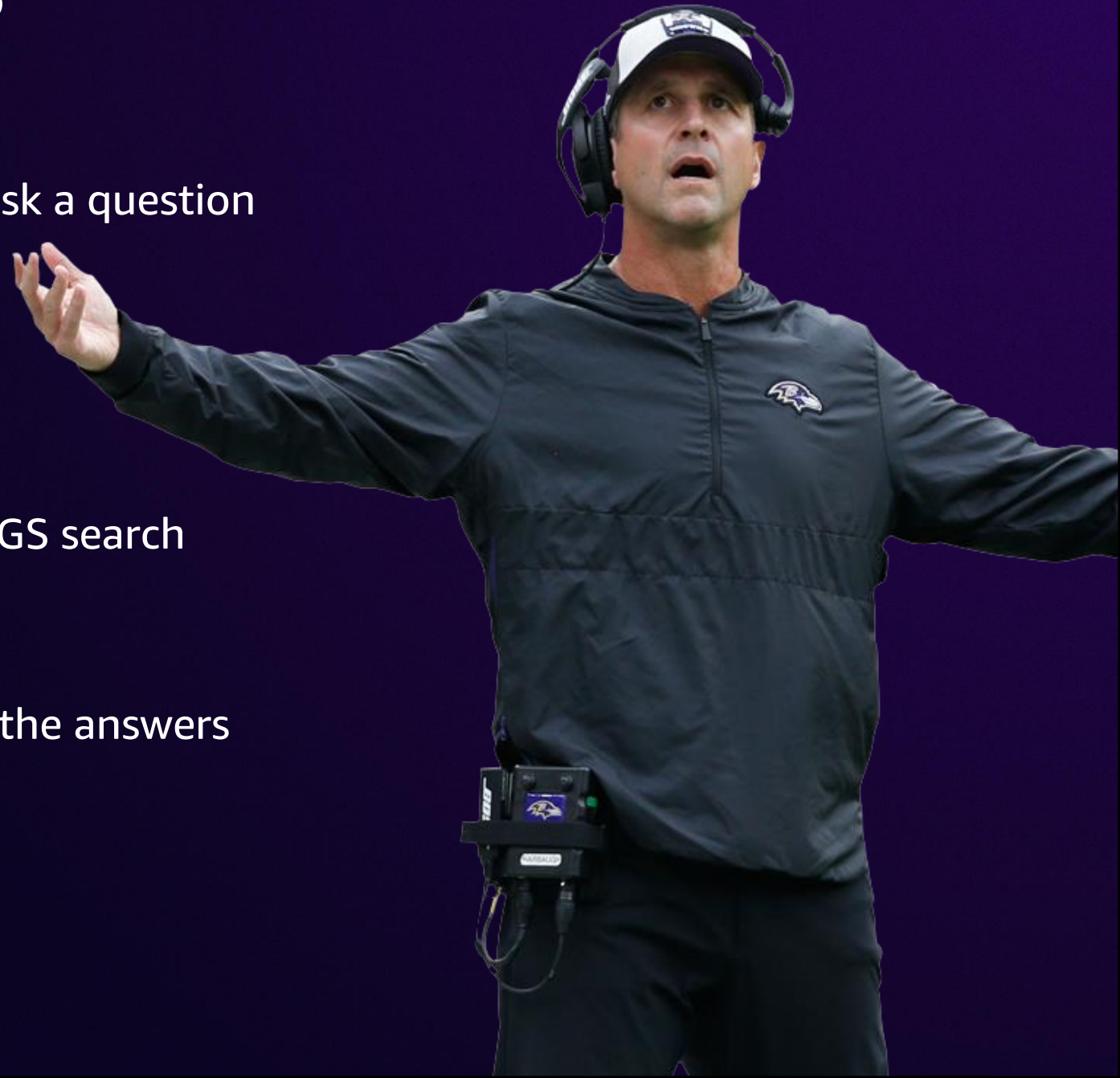
Problem – How can we search more efficiently?



- Fast-paced live production environment not conducive to lengthy & complicated search
- Internal Next Gen Stats research tools are robust but require manual effort & time
- Setting up filtered search & getting to the resulting media that matches can take 10 minutes for a list of 20 plays
- Not just looking for a single shot from a single play – iterative process so search time stacks quickly

Why use generative AI?

- What if you could use natural language to ask a question of Next Gen Stats?
- Natural language query → NGS API call
- Barrier to entry with performing complex NGS search would be reduced
- Conversational refinement of search to get the answers user is looking for



Build generative AI MVP with NFL



Diagnostic process

Introductory call to learn more about use case overview and business needs and to identify stakeholders



Discovery workshop

Deep dives to understand existing solutions, challenges, prioritize features



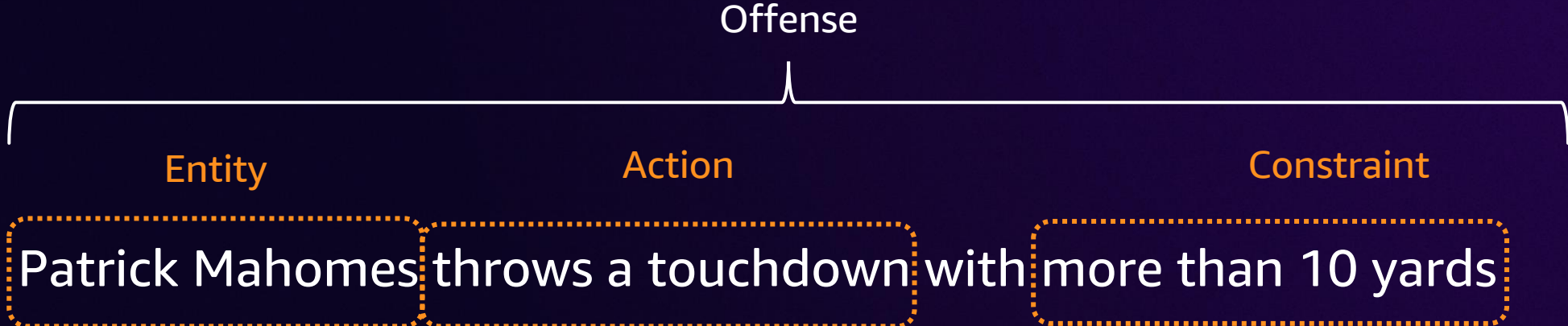
POC execution phase

Recommend execution road map and develop a proof of concept

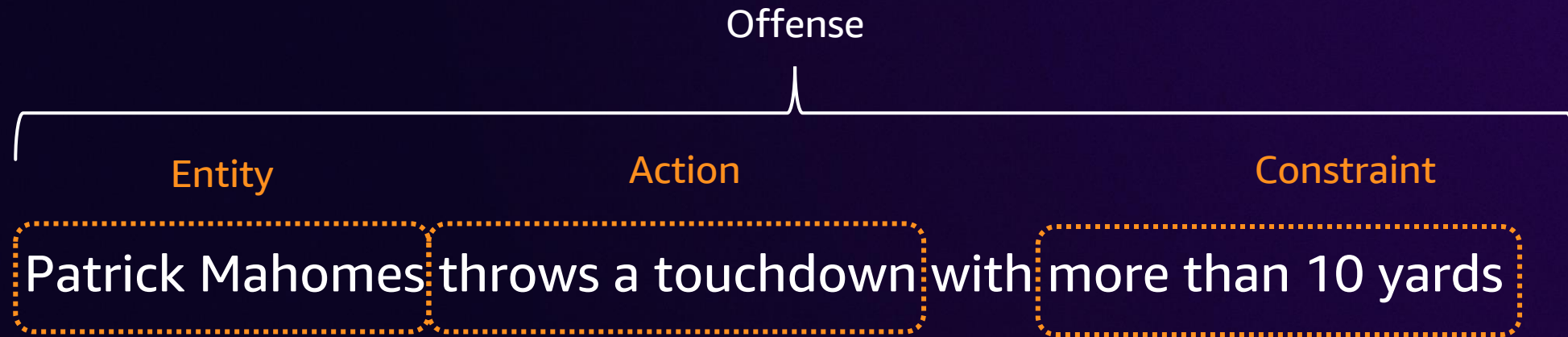
Problem formulation

- **Business objective:** query content easier and faster
- **Challenges:**
 - Translate arbitrary user requests – E.g., “Find me all the plays where Patrick Mahomes throws a touchdown with more than 10 yards this season”
 - Do it accurately
 - Do it efficiently

Problem formulation



Problem formulation



API Filters: `passerid=44822 & pass_touchdown=1 & pass_passYards=>10`

Identify relevant entities

- **Requirement:** associate IDs for entities mentioned in the query (players/teams)
- **Challenges:**
 - Users may search only through last name or acronyms, sometimes can contain typos
 - Some players share the same names
- **Solutions:**
 - Built custom matching algorithm for efficient lookup that handles typos, last name, and acronym searches
 - Clarify with users if “duplicated names” found

Identify relevant actions and constraints

- **Requirement:** generate correct API calls to be executed
- **Challenges:**
 - Language model doesn't know how to link API fields with actions & constraints
 - Hard to get all conditions for complex user queries
- **Solutions:**
 - Ingest corresponding documentations for models to "learn the rules"
 - Ask model to think step by step and include reasoning as output
 - Include correct and incorrect examples to "further instruct" language models

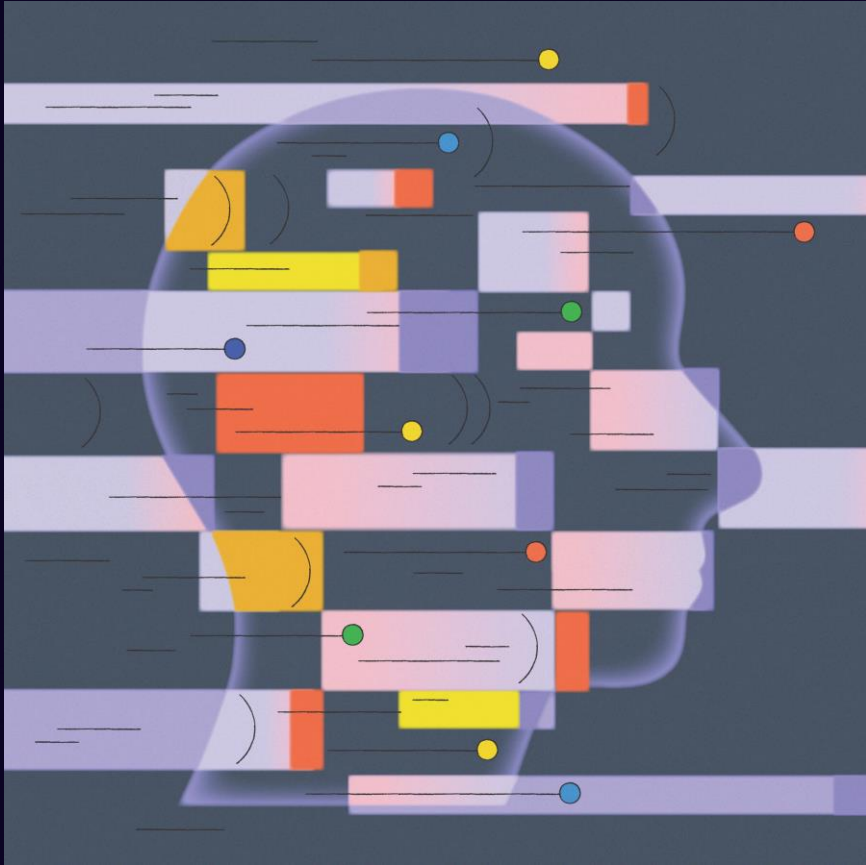
Performance evaluation

- **Requirement:** measure accuracy and latency of the solution
- **Challenges:**
 - Lack of ground truth labels
- **Solutions:**
 - Domain experts generate ground truth samples of varying difficulty
 - Human in the loop evaluations in iterative loops
- **Outcomes:**
 - Achieved 90% accuracy on user acceptance test
 - Average response time is about 20 seconds

Remaining challenges

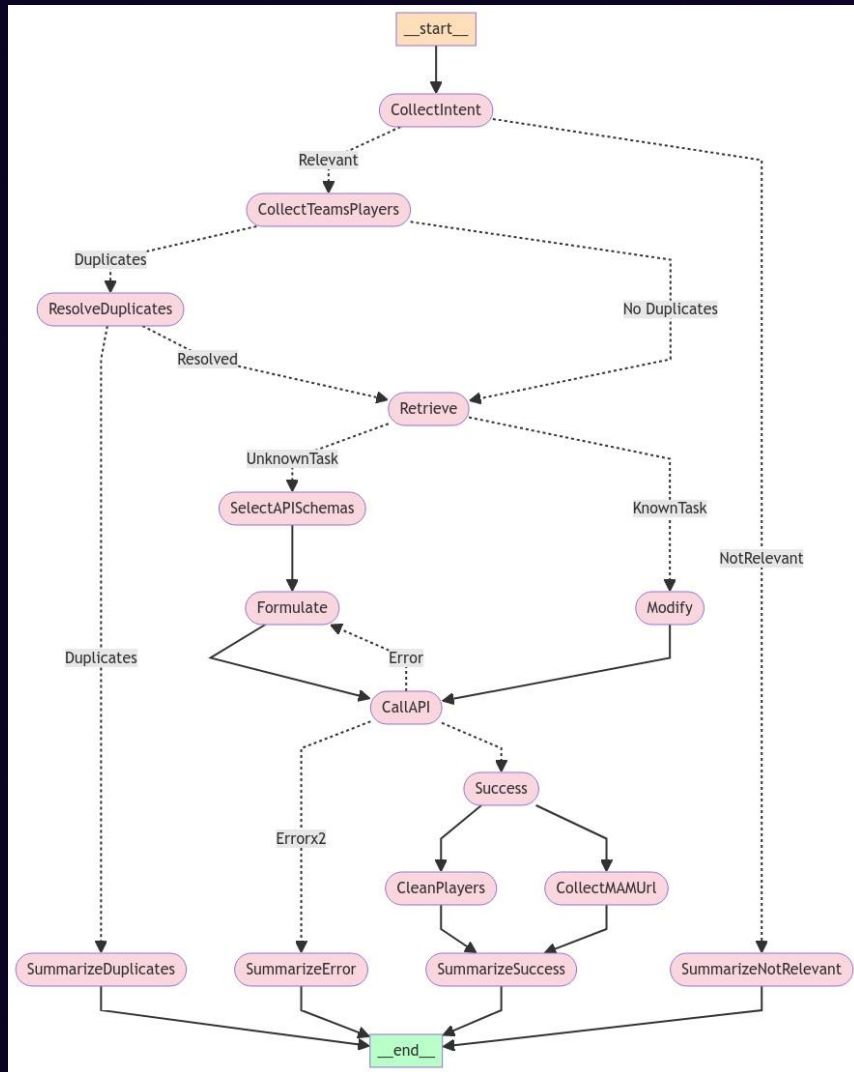
- How can we provide a more seamless conversational experience?
- How can we link extracted play information with media assets?
- How can LLM leverage user feedback for enhanced searching?
- How can we further reduce model hallucinations?

Goal – Mimic human decision-making



- Leverage and create memories
- Break larger problem into smaller simpler problems
- Reach a conclusion through logical sequence of outcomes

Solution – Agentic workflow



Graph-based architecture with **nodes**, **edges**, and **state**

- Node: Any function performing an LLM call, vector search, etc.
- Edge: relationship between nodes
- State: passed from node to node

Assess user intent

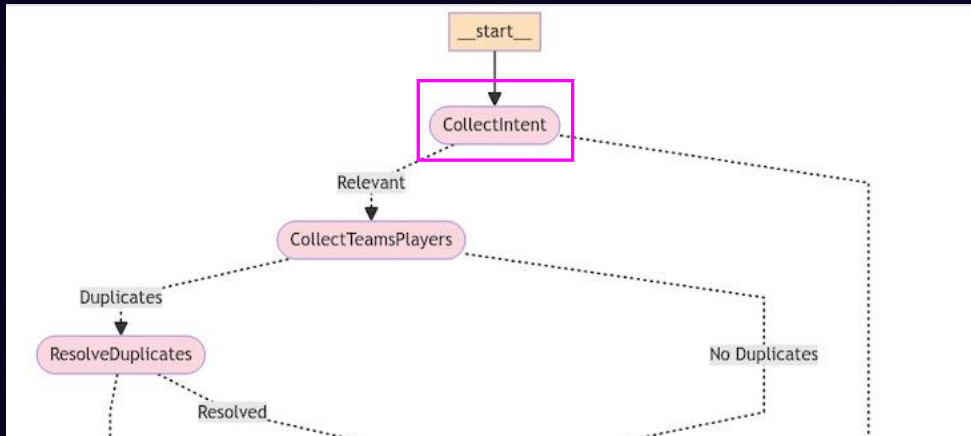
Node input:

“Find me all touchdowns Brock Purdy has thrown against the Cardinals”

“How about just in the fourth quarter?”

Node output:

“Find all touchdowns Brock Purdy has thrown against the Cardinals in the fourth quarter.”



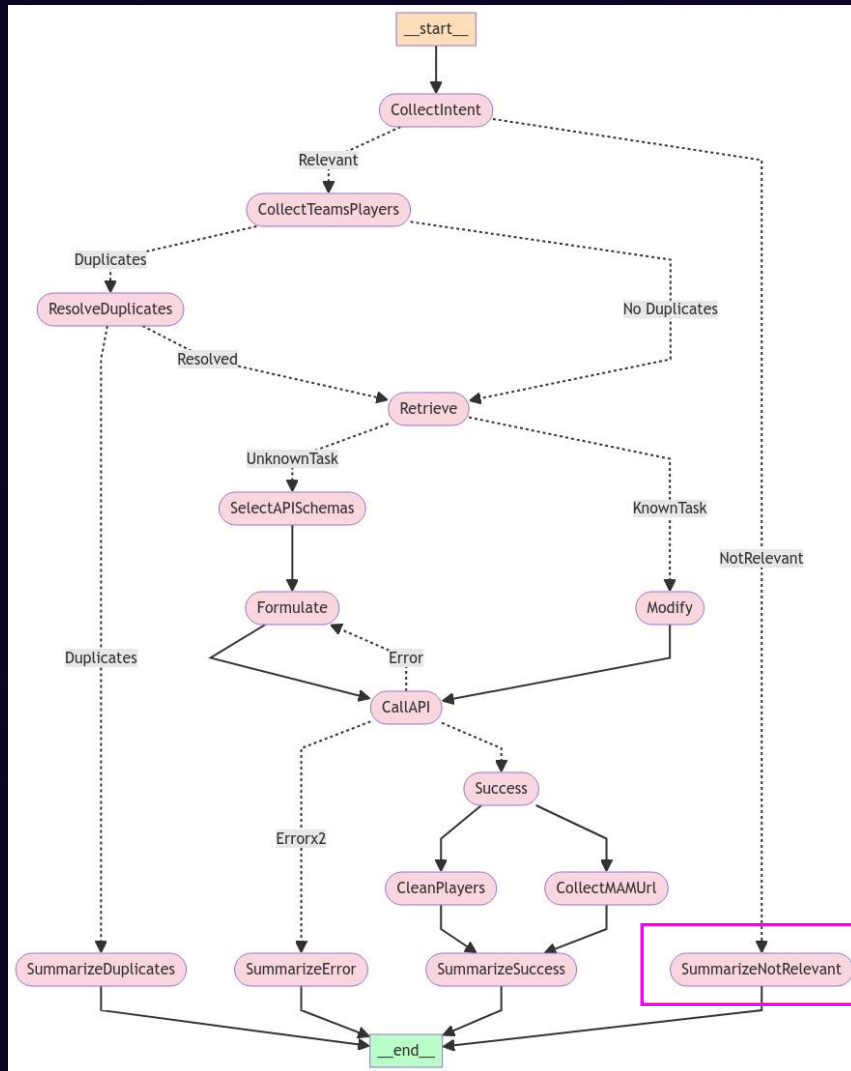
Assess user intent – Relevance

Node input:

“What was the best movie made in 2023?”

Node output:

“I’m sorry, I am only designed to answer specific questions related to ...”



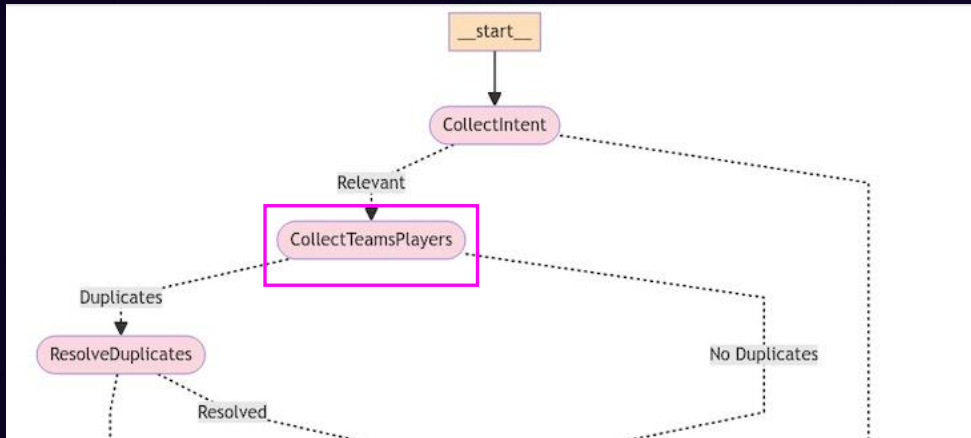
Identify teams and players

Node input:

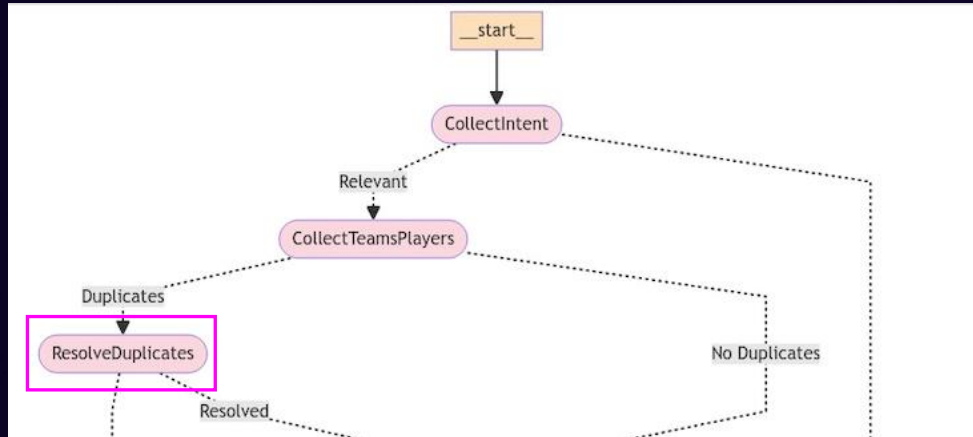
“Find me all touchdowns Brock Purdy has thrown against the Cardinals”

Node output:

```
{  
  "players": [{"name": "Brock Purdy",  
                "position": "QB",  
                "team": "SF",  
                "id": 1234}],  
  "teams": [{"name": "Arizona Cardinals",  
             "id": 5678}]  
}
```



Attempt to resolve duplicate player names



Node input:

"Find me all touchdowns scored by Michael Carter

Here are the available players:

<player detail>

"

Node output:

Assessment of correct player based on input

Request clarity from user

User input:

“Find me all touchdowns scored by Michael Carter”

Node output:

“I have found multiple players – can you confirm?”

Player Name: Michael Carter

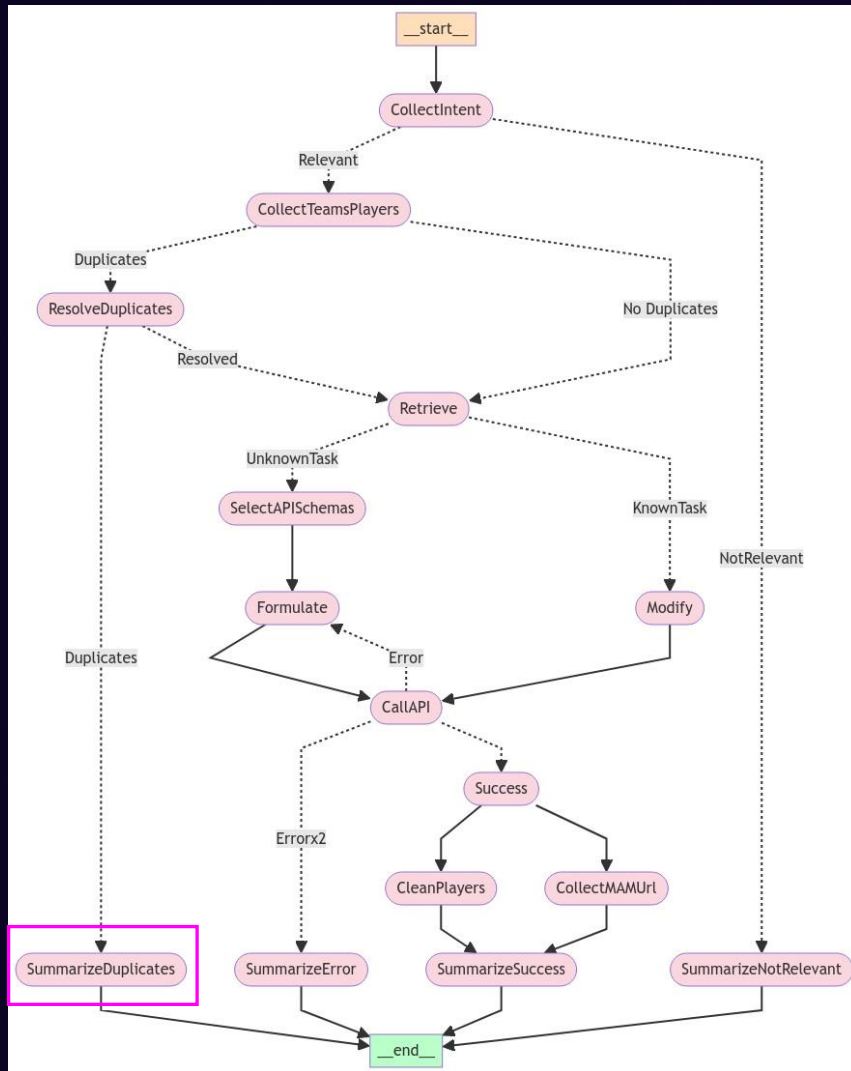
Team: Jets

Position: RB

Player Name: Michael Carter

Team: Jets

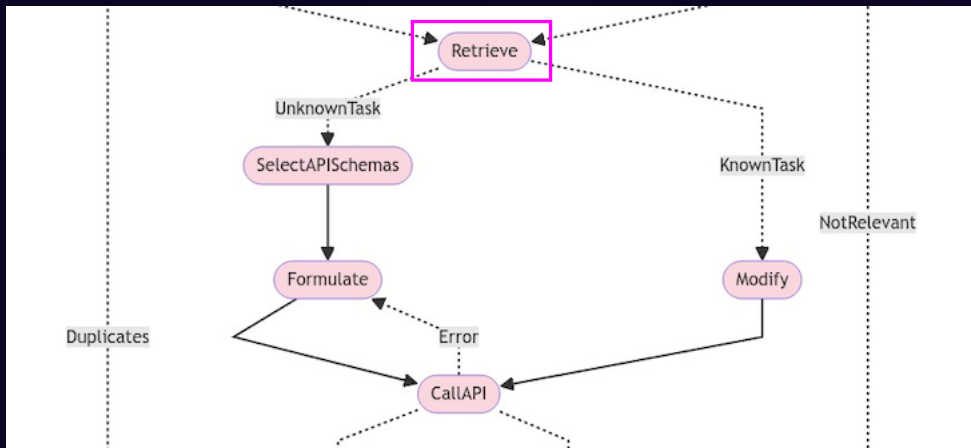
Position: DB”



Retrieve similar task from memory

Node input:

“Find all touchdowns Brock Purdy has thrown against the Cardinals in the fourth quarter.”



Node output:

1. Most similar task from memory
2. API call used to collect information for this task
3. Though process behind the API call
4. Distance between search and nearest match

Select relevant API documentation

Node input:

“Choose from the available API documentation:

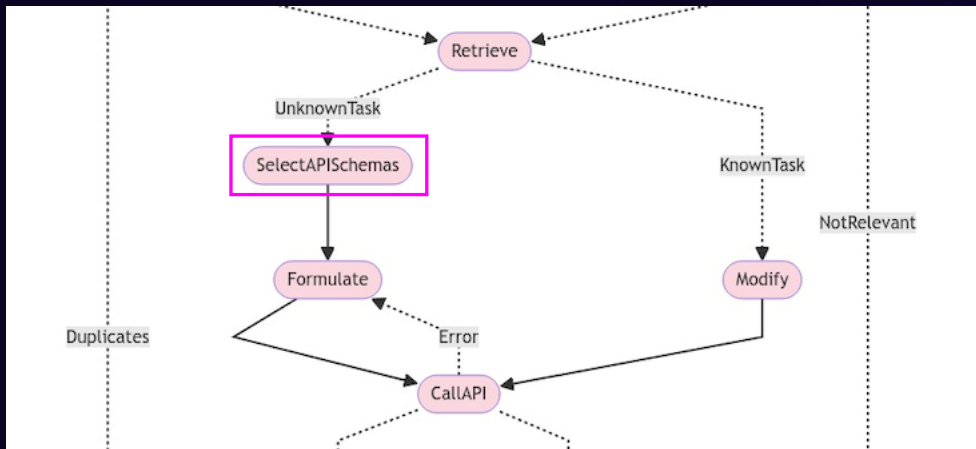
Defensive formations – <definition>

Passing plays – <definition>

Rushing plays – <definition>

...”

Node output: “Passing plays and general offense”



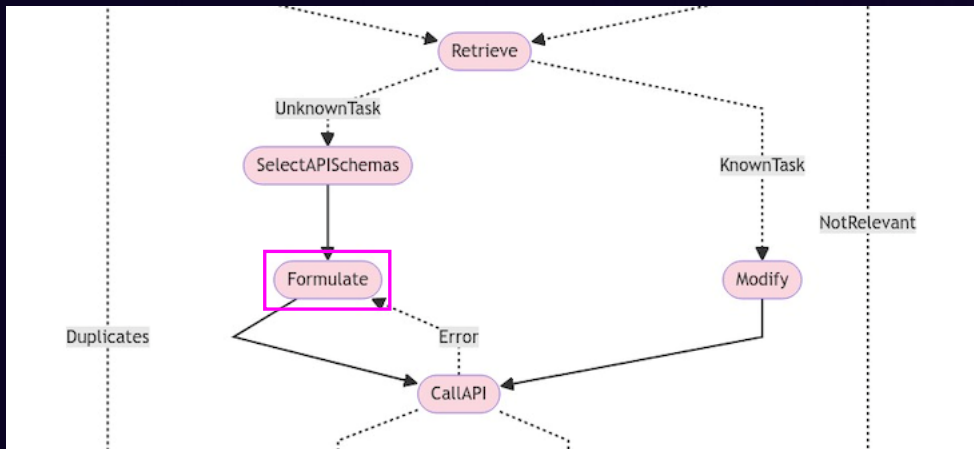
Formulate an API call

Node input:

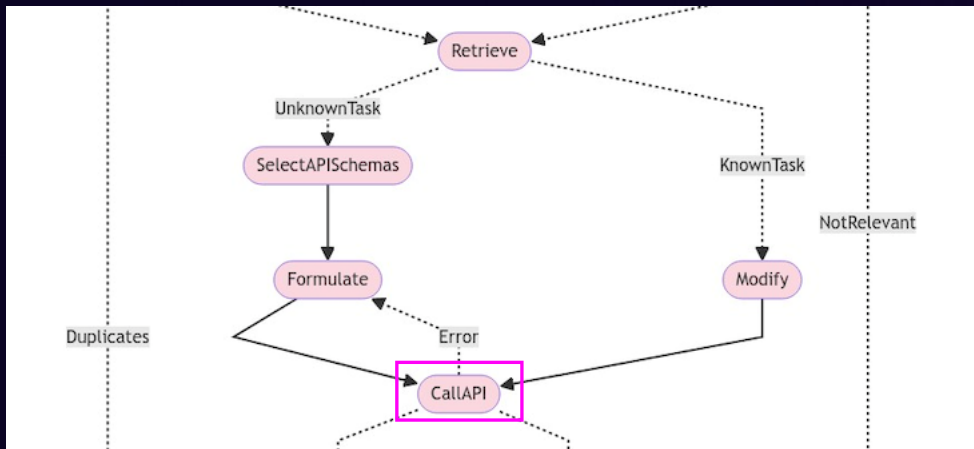
“Review the API documentation below and formulate an API call to meet this request”

Node output:

1. Proposed API call to collect information for this task
2. Thought process behind the API call



Execute the API call



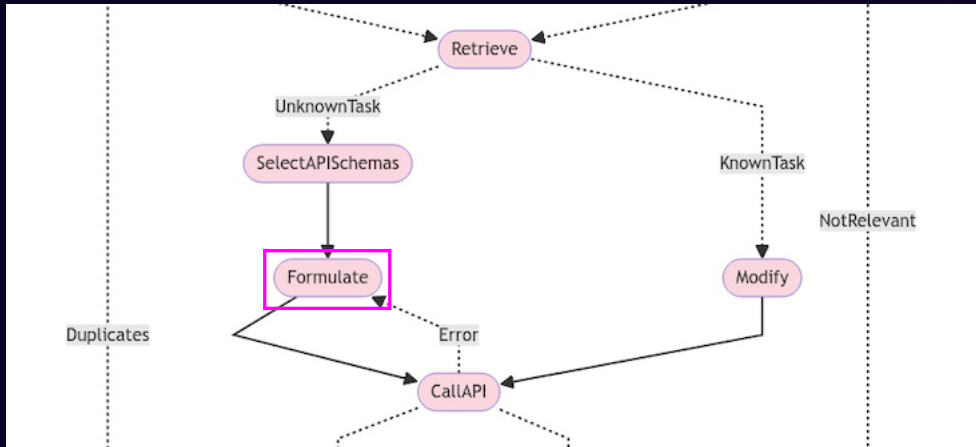
Node input:

The proposed API call needed to collect the desired information

Node output:

Formatted API output

Error handling – Reformulate API call



Node input:

“You have already attempted to formulate an API call for this request.”

Here was the request: <>

Here was the error thrown: <>

Update this API call to address the error”

Node output:

1. Updated API Call to collect information for this task
2. Thought process behind the API call

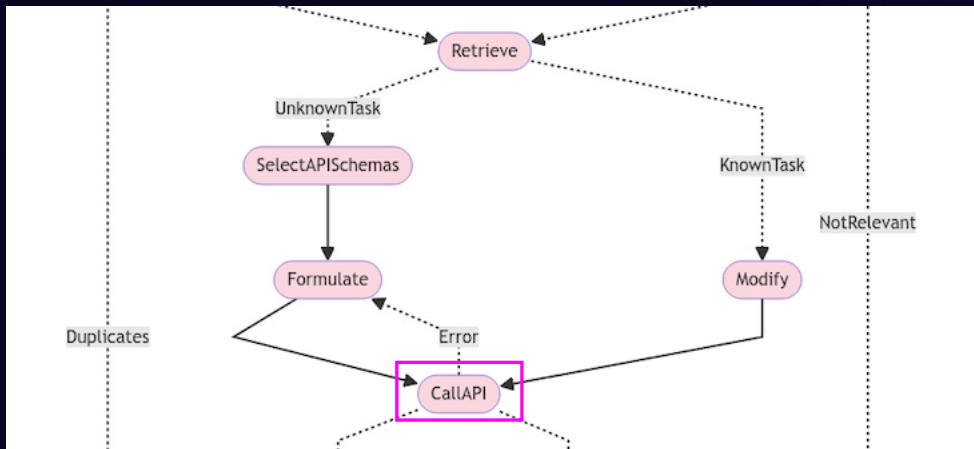
Execute the updated API call

Node input:

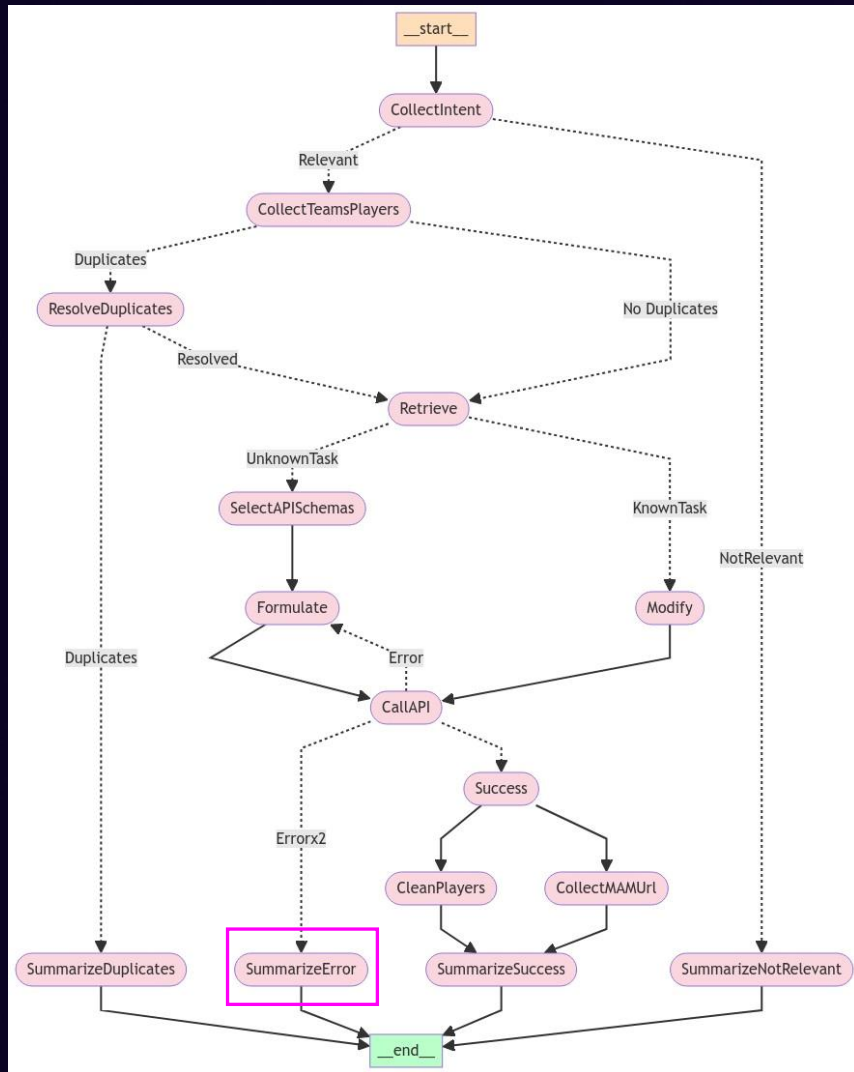
The updated API call needed to collect the desired information

Node output:

Formatted API output



Error x2: Request clarification



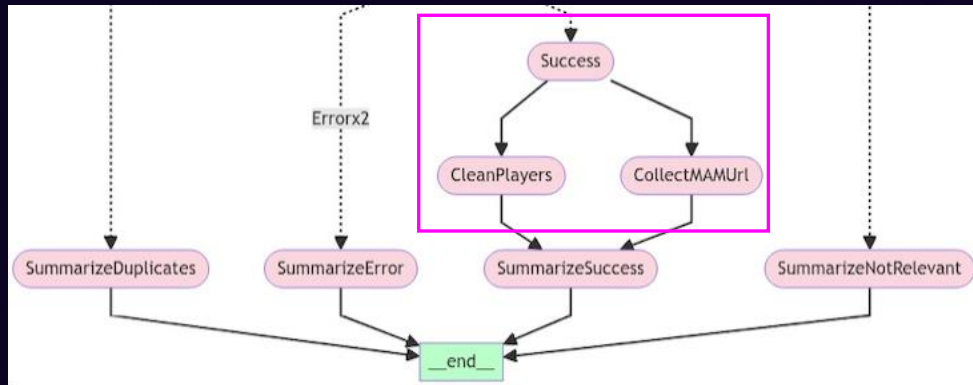
Node output:

"I have attempted to make the following API call:
<>

Here is the error message thrown: <>

Can you please restate your request?"

API success – Query media asset management



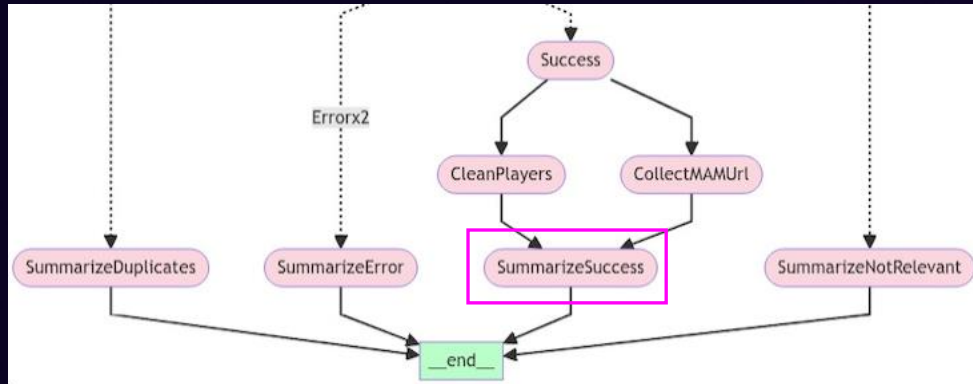
Node input:

Game and play IDs from Next Gen Stats

Node output:

Links to plays in the media asset management system

Summarize successful response



Node input:

All information collected from Next Gen Stats and media asset management

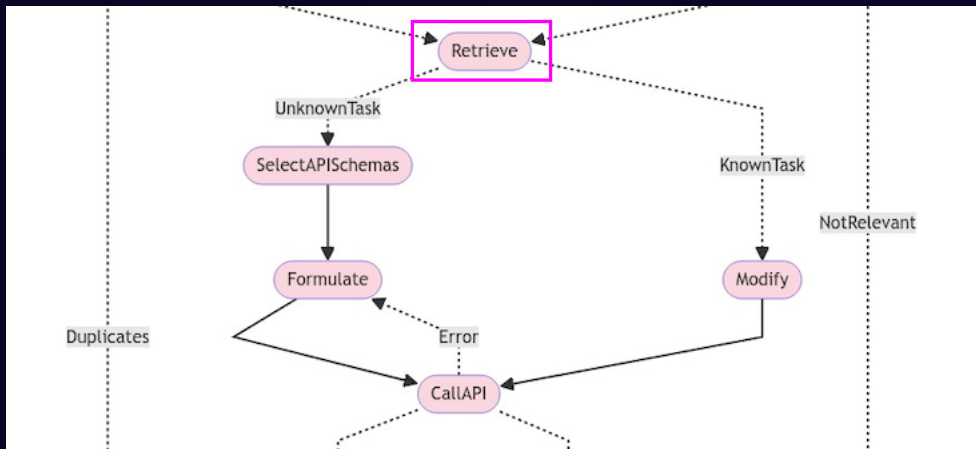
Node output:

1. Streaming summary of the plays returned
2. Formatted table with play metadata and links to media assets

Retrieve similar task from memory

Node input:

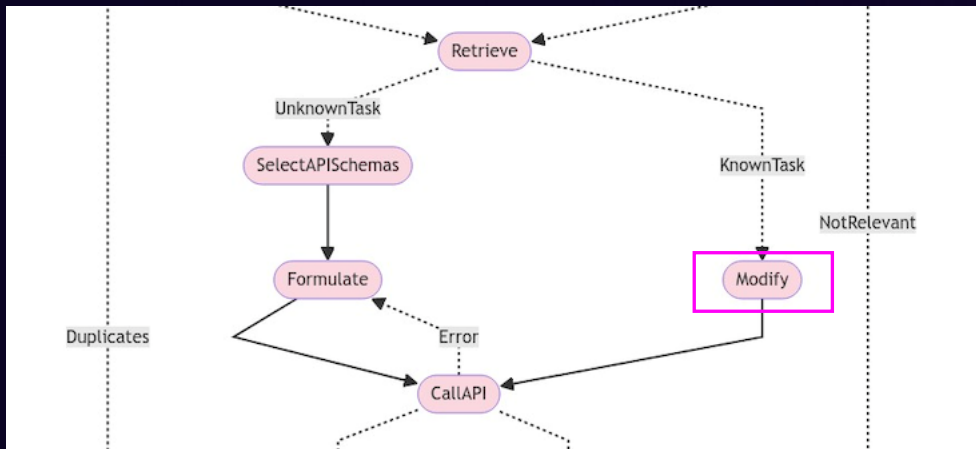
“Find all touchdowns Brock Purdy has thrown against the Cardinals in the fourth quarter.”



Node output:

1. Most similar task from memory
2. API call used to collect information for this task
3. Though process behind the API call
4. Distance between search and nearest match

Modify a known task



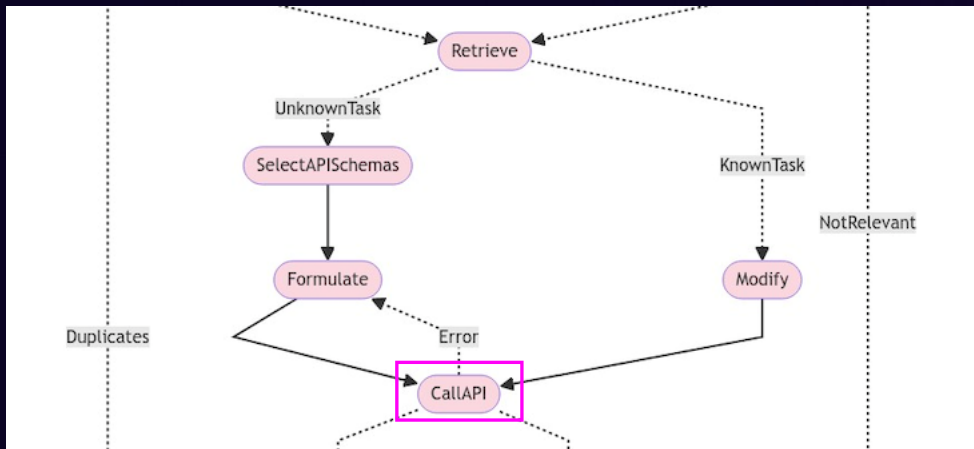
Node input:

“Modify this API call to collect information for this similar task.”

Node output:

1. Updated API call to collect information for this task
2. Thought process behind the API call

Execute the API call



Node input:

The proposed API call needed to collect the desired information

Node output:

Formatted API output

Updating long-term memory

The screenshot shows a chat window with a dark theme. At the top right, there is a user profile labeled 'ADMIN'. A blue bubble contains the user's request: 'Justin Jefferson receptions in the 4th quarter when his team is trailing'. Below this, the chatbot's response is displayed with a profile picture of a football player. The response is structured as follows:

User Request:
Justin Jefferson receptions in the 4th quarter when his team is trailing

Summary:
Based on the API results provided, there were no recorded receptions by Justin Jefferson in the 4th quarter when his team was trailing during the 2024 regular season. The API call was successfully executed, but it returned zero game plays matching the specified criteria, indicating that Jefferson did not make any such receptions in the given timeframe and game situation.

Highlights:

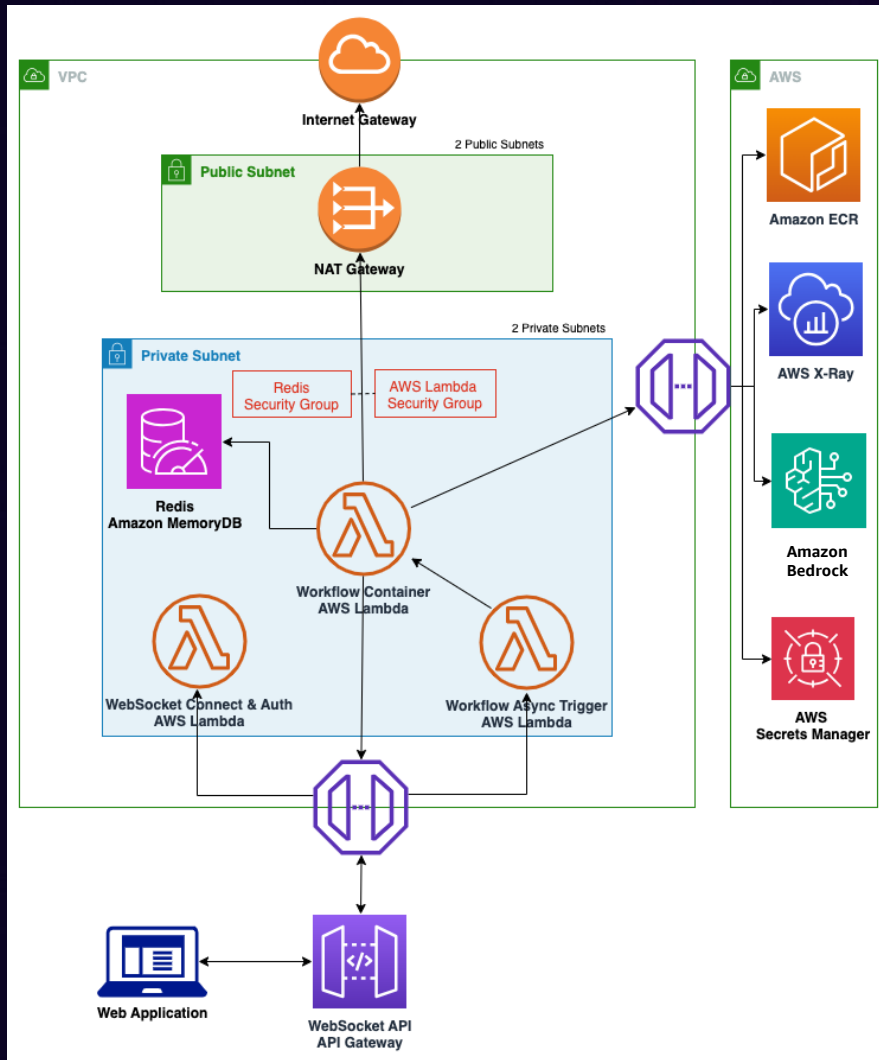
- No receptions recorded for Justin Jefferson in the specified scenario
- API call targeted the 2024 regular season
- Focus was on 4th quarter plays when Jefferson's team was trailing
- Successful API execution with zero matching results
- Data suggests Jefferson had no receptions in these specific game situations

Wrap-up:
The API results indicate that Justin Jefferson did not record any receptions in the 4th quarter when his team was trailing during the 2024 regular season, suggesting either limited opportunities or unsuccessful targets in such situations.

Please let me know if you would like for me to make any changes or updates.

At the bottom, there is a blue input field with the placeholder text 'Message NFL Chatbot...' and a send button on the right.

Bringing it all together



Primary infrastructure components:

- AWS Lambda
- Amazon MemoryDB
- Amazon Bedrock
- WebSocket API

Primary conceptual components:

- Language models
- Workflow orchestration
- Memory

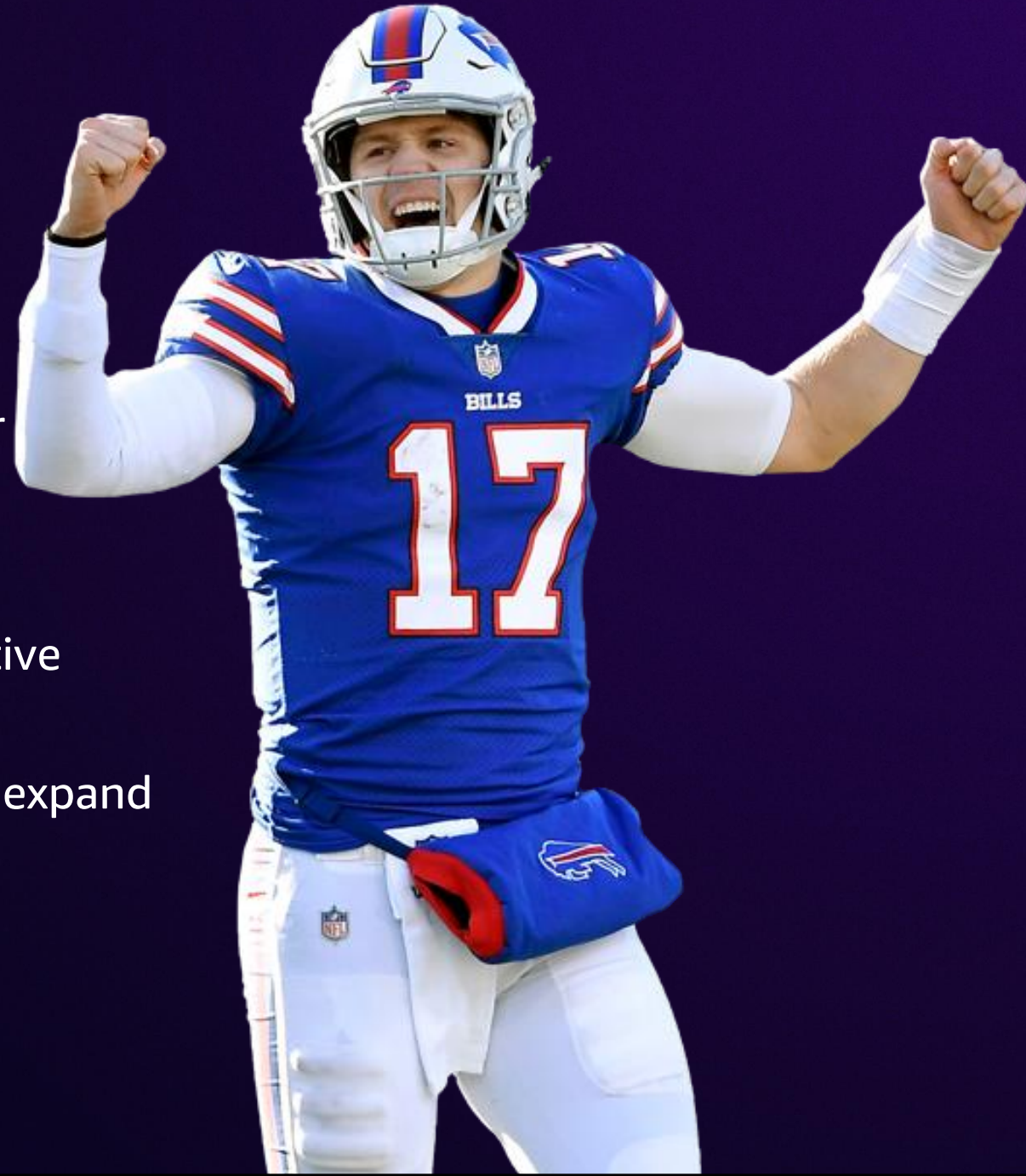
Keys to a successful project

- Quickly get to MVP
- Onboard select alpha testers early to start finding edge cases
- Get a version into the hands of end users as quickly as possible
- Iteratively incorporate feedback to continually enhance performance
- Test & retest & retest



Impact of Playbook Pro

- In production as of August 2024
- 250 users onboarded in first month
- 30 seconds to have all angles of your plays at your fingertips instead of 10 minutes
- More time for reviewing footage options
- Content creators can spend more time in the creative space, less time getting there
- Immediately started thinking about where we can expand the functionality to next



What's next for Playbook Pro?

- Just the tip of the Next Gen Stats iceberg; many more API endpoints we could be leveraging to more deeply integrate with NGS
- Expand the range of APIs we are leveraging beyond just NGS
- How can we take the learnings from this internal tool to create something fan facing?
- Many directions we can go next



Thank you!

Eric Peters



Henry Wang



Ross Claytor



Please complete the session survey in the mobile app