Agentic Generative AI for Media Content Discovery at the National Football League

Henry Wang[1], Md Sirajus Salekin[1], Jake Lee[1], Ross Claytor[1], Shinan Zhang[1], and Michael Chi[2]

[1] Amazon Web Services, USA
[2] National Football League, USA

Abstract. Generative AI has unlocked new possibilities in content discovery and management. Through collaboration with the National Football League (NFL), we demonstrate how a generative-AI based workflow allows media researchers and analysts to query relevant historical plays using natural language, rather than using traditional filter and click-based interfaces. The agentic workflow takes a user query in natural language as an input, dissects the query into different elements, and then translates these elements into the underlying database query language. The accuracy and latency of retrieval are further improved through carefully designed semantic caching. The solution performs with over 95-percent accuracy and reduces the average time of finding relevant videos from 10 minutes to 30 seconds, significantly increasing the NFL's operational efficiency and allowing users to focus more on producing creative content and engaging storylines.

Keywords: National Football League · Next Gen Stats · Agentic System · Natural Language Query · Media Content Search

1 Introduction

The demand for sports content grows every year, as shown by the increasing sizes of sports leagues' multi-year media rights deals valued at billions of US dollars [2, 15]. The National Football League (NFL)[3], one of the most popular professional sports leagues in the world, produces and distributes content all year-round to meet this demand and to engage its 184 million domestic and 100 million international fans. The NFL manages petabytes of content, primarily videos of game footage, and distributes it for use across broadcasts, TV segments, digital platforms (sites, apps), and social media. In a sports entertainment landscape that is becoming increasingly more digital, producing captivating content remains at the core of what the NFL delivers to fans around the world. However, it has become increasingly difficult for media teams to find and manage the right content in the NFL's ever-growing library of digital media.

The NFL partnered with our team to develop an agentic solution - powered by generative AI - that enables media and production teams to efficiently search

[3] https://www.nfl.com/

for game videos using natural language queries[4]. The solution enables users to spend more time on the creative aspects of their roles to create more engaging content for fans.

2 Related Works

The advent of generative AI has revolutionized traditional content retrieval paradigms, transforming how we interact with and extract information from vast data repositories. With the power of Large Language Models (LLMs) [23], we can now perform text-to-SQL and Retrieval-Augmented Generation (RAG) operations that convert natural language queries into corresponding intermediate representations of the SQL queries, vector embeddings, or other structured formats, thus, enabling more intuitive and effective content retrieval. Apart from the general in-context learning capability of LLMs, recent text-to-SQL approaches can solve different complex text-to-SQL tasks using chain-of-thoughts [14], few-shot strategy [17], decomposing steps [18], self correction [20], and multi-turn steps [21]. On the other hand, RAG [3] frameworks can retrieve relevant context for any particular query. Both text-to-SQL and RAG systems can retrieve domain-specific content based on SQL or vector embeddings.

In sports, media content retrieval is a common task when preparing highlights or writing digital posts. Recently, LLMs have shown their potential in transforming how sports organizations manage, analyze, retrieve, and discover content within their vast media archives. LLMs are being used for applications including: expert commentary generation [8, 1, 4, 9], action spotting [12], sports understanding [16, 19, 6, 10, 7, 22], and video assistant refereeing systems [5]. Natural language queries are making sports content more accessible to analysts, fans, and broadcasters. Related work in soccer (football) has shown rather than relying on structured search parameters or specific keywords, users can now query and retrieve soccer information using RAG [8, 13] or GraphRAG [11].

While LLMs introduce the capability to perform natural language queries and to retrieve relevant content, the extent to which LLMs can be utilized to understand and streamline complex data and media platforms (such as the NFL's Next Gen Stats platform) is yet to be explored. Specifically, it would be worthwhile to explore whether LLMs and natural language prompts can be utilized to automate and streamline the complex backend API calls used to retrieve desired media content.

3 Data

The NFL's Next Gen Stats (NGS) platform[5] is a comprehensive player and ball tracking system that captures and stores real-time data on every play from

[4] https://www.wired.com/sponsored/story/will-the-nfls-push-into-genai-transform-how-we-see-sports/

[5] https://nextgenstats.nfl.com/

every NFL game since 2016. It stores large volumes of information, including information on players, teams, historical performances, and relevant advanced statistics. This data is captured through sensors in players' pads and the ball.

To build and test our agentic media search solution, our team used different statistical data such as team/player/play details, glossaries, question-answer pairs for media search, and backend OpenSearch API call syntax from the NGS platform. Our team also worked closely with NFL Subject Matter Experts (SMEs) to prepare and validate the ground truths and to retrieve search results during solution development.

3.1 Schema

Due to highly granular nature of the NFL's NGS data, the data is organized into different schemas (such as defense vs offense, passing vs rushing) to help with extracting stats and facilitating data organization. The NGS data is stored in OpenSearch and each schema consists of its own unique OpenSearch keys and fields. Table 1 & 2 (Appendix) contains descriptions of key fields to demonstrate what the schemas look like. These schemas are passed into LLMs as context to help them understand the data types and definitions of each field.

3.2 Question and Answer Pairs

240 Question and Answer (QA) pairs were collected to represent user queries and their ground truth number of plays. The questions varied in complexity and were representative of questions that would be typically asked by NFL's target end users such as research and production analysts - 30% are easy questions that can be answered by straightforward API calls, 50% are medium questions that require multiple filtering conditions and the remainder 20% are complex questions that not only require deep football knowledge but also may involve multiple schema. 100 of the questions were set aside for the test set and the rest were used as development data to construct few-shots examples and enhance our prompting strategy. Examples of the QA pairs are shown in Table 3 (Appendix).

4 Methodology

We built an end-to-end, natural-language "agentic" search experience - the system interprets a user's query, asks clarifying questions when necessary, then composes and executes OpenSearch calls to fetch the matching plays and their media links. The workflow runs in AWS, is orchestrated with LangGraph and is surfaced through a React front-end (screenshots in the Appendix). Figure 1 gives the high-level flow. Details of the design are described below.

4.1 Agentic Workflow

The distinctly different schemas warrant the need for an agentic workflow that interprets which tables and fields are needed when constructing the API calls.

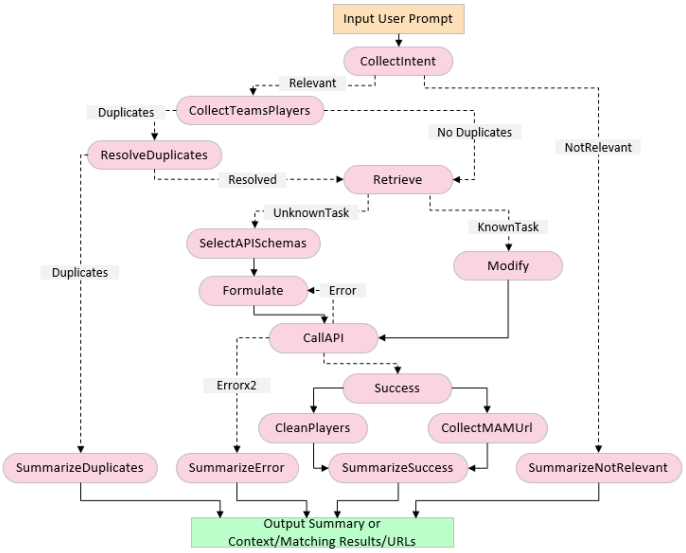

Fig. 1. Overall diagram of the Agentic workflow

Assess user intent A low-latency model (such as Claude 3 Haiku) classifies the prompt. Non-football questions receive a polite rejection; valid football queries trigger the graph. This guardrail prevents out-of-scope inputs from derailing later steps.

Extract entities, actions, and conditions The LLM breaks down the prompt into Entities (players/teams), Actions (stats of interest), and Conditions. For a query such as "*Find all plays where Patrick Mahomes throws a touchdown farther than 10 yards*," "Patrick Mahomes" is the Entity, "touchdown throw" is Action, ">10 yards" is the Condition. When situations with duplicate names appear (it's not uncommon for players in NFL to have same last names), the LLM prompts the user to confirm their player of interests and store confirmed player/team id for the session.

Select relevant API schema: Based on extracted entities, actions and constraints, the system first conduct a semantic similarity search to identify if similar queries have been encountered in the past:

- If not, the system will pin down relevant context. A router LLM picks the most relevant NGS schema(s)—e.g. *passing*, *rushing*, *team defense*, *team offense*, etc. Each schema contains relevant fields that LLM can use to construct the API call. For example, the "*passYards*" field corresponds to the yardage of passing, and "touchdown" is a boolean indicator of whether the play resulted in a touchdown. Only the relevant schemas are passed in as context to reduce overall context length and reduce the chance LLM picks inaccurate fields.

– If similar queries are addressed before, the system will directly leverage the information instead of reformulating the problem. Details are described in **Semantic caching** below.

Formulate and execute API calls: The API-formulation LLM receives team/player IDs, reference schema, and few-shot demonstrations on how APIs are constructed. It reasons step-by-step, maps extracted actions and conditions to corresponding fields and emits the final API call with values assigned to each field. Figure 2 from Appendix shows a sample of formulated API call and reasoning LLM provides. A python runner executes the API call and an LLM summarizes the returned results into natural language responses to the user; Upon syntax failure, the LLM will attempt auto-correct using error messages. If the final API call still fails after three tries, the model will prompt the user to rephrase original query and run the workflow again.

Conversational experience: Follow-up conversations inherit prior context. After "*Find me all the plays where Patrick Mahomes throws a touchdown farther than 10 yards*," a user may ask "*What about all the throws that were intercepted?*" The LLM implicitly keeps "Patrick Mahomes" as the entity, and will rephrase the user's prompt to "*Find all the plays where Patrick Mahomes throws are intercepted.*"

Semantic caching: We optimize latency and accuracy by storing prior queries and responses with player/team names redacted and replaced by placeholders (e.g. "[PLAYER]", "[TEAM]"). When a new query gets submitted, the system first applies a vector search to find similar queries in the past. If a high score is achieved (meaning highly-similar queries have been addressed in the past), the cached API calls with placeholders will be replaced with the fresh IDs and executed directly, allowing the system to bypass expensive reformulation.

Link plays to assets in Media Asset Management(MAM) solution: Finally, the play IDs are serialized and sent to NFL's MAM system, which will retrieve unique URLs for each asset. The React UI embeds the returned URLs. See Figure 2 for an example where a user can click on any of the returned NGS Media Links to access the media content directly in a MAM. At this point, the end-to-end content searching experience with natural language is complete.

4.2 System Infrastructure

The entire agentic workflow is implemented in the AWS ecosystem. There are 3 major components of this system. These are: 1) LLM model using Amazon Bedrock converse API, 2) Memory using Redis and 3) Workflow orchestration using LangGraph. The overall architecture diagram of the infrastructure is included in Appendix Figure 5.

5 Results

To develop the solution, our team used 100 QA pairs to refine our prompts and iteratively collected feedback from NFL SMEs to create a smooth user experience. The answer is considered accurate if the constructed APIs contain the

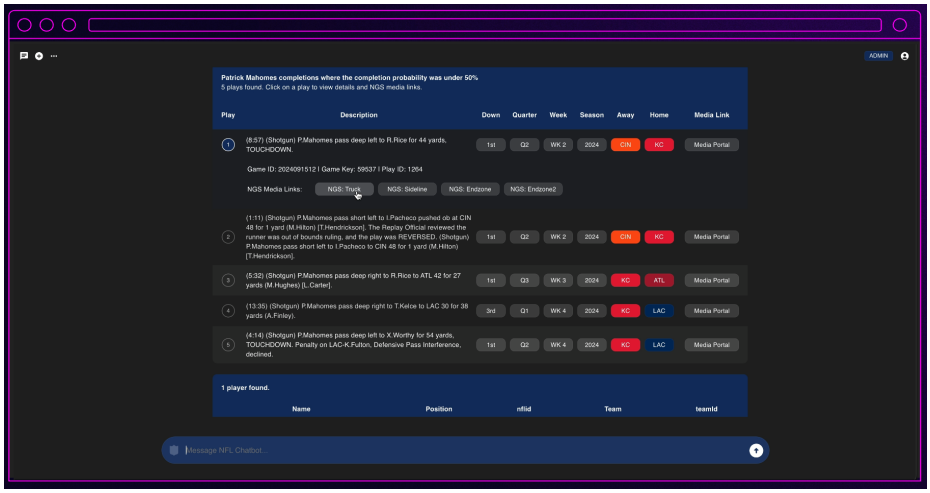

Fig. 2. Links to retrieved assets embedded in React frontend

correct filter&values and the count of records matches with ground truth value. During testing, the solution achieved over 95% accuracy on the 100 QA pairs. The solution has yielded notable process improvements across the NFL's media teams and 250 users were onboarded within the first month after its deployment. Users reported significant time savings: With free-form natural language queries, users can now search for and retrieve relevant video in an average of 30 seconds, down from over 10 minutes per query, giving them more time to focus on finding creative insights. They can also ask follow-up questions if they deem certain results returned interesting, so they can maintain the research flows without navigating between different tools.

6 Conclusion

In this paper, we discussed the implementation and efficacy of LLM-driven agentic workflows in transforming natural language queries into complex API calls within the NFL's Next Gen Stats system. Today, this solution is used daily by all of the NFL's media editors. For simple searches that are expected to return videos for only a few specific plays, users can find relevant videos nearly 6 times faster than before. This time savings for finding relevant videos scales up to 20 times faster as search complexity increases and more game plays match the user's prompt. In the future, there are opportunities to further enhance the capabilities of the tool and create fan-facing version of the solution, providing richer contents and offerings to hundreds of millions of avid football fans worldwide.

References

1. Cook, A., Karakuş, O.: Llm-commentator: Novel fine-tuning strategies of large language models for automatic commentary generation using football event data. Knowledge-Based Systems **300**, 112219 (2024)
2. Elberse, A., Warner, E.: The nfl's $110-billion media rights deals. https://www.hbs.edu/faculty/Pages/item.aspx?num=62434 (2022), accessed: 2025-04-17
3. Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.S., Li, Q.: A survey on rag meeting llms: Towards retrieval-augmented large language models. In: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 6491–6501 (2024)
4. Ge, K., Chen, L., Zhang, K., Luo, Y., Shi, T., Fan, L., Li, X., Wang, G., Zhang, S.: Scbench: A sports commentary benchmark for video llms. arXiv preprint arXiv:2412.17637 (2024)
5. Held, J., Itani, H., Cioppa, A., Giancola, S., Ghanem, B., Van Droogenbroeck, M.: X-vars: Introducing explainability in football refereeing with multi-modal large language models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3267–3279 (2024)
6. Jiang, T., Wang, H., Salekin, M.S., Atighehchian, P., Zhang, S.: Domain adaptation of vlm for soccer video understanding. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 6111–6121 (2025)
7. Lee, C., Lin, T., Pfister, H., Zhu-Tian, C.: Sportify: Question answering with embedded visualizations and personified narratives for sports video. IEEE Transactions on Visualization and Computer Graphics (2024)
8. Li, X., He, Y., Zu, S., Li, Z., Shi, T., Xie, Y., Zhang, K.: Multi-modal large language model with rag strategies in soccer commentary generation. In: 2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 6197–6206. IEEE (2025)
9. Sarfati, N., Yerushalmy, I., Chertok, M., Keller, Y.: Generating factually consistent sport highlights narrations. In: Proceedings of the 6th International Workshop on Multimedia Content Analysis in Sports. pp. 15–22 (2023)
10. Schilling, A., Anurathan, J., Mühlberger, J., Gerschner, F., Rössle, M., Theissler, A., Klaiber, M.: Querying football matches for event data: Towards using large language models. In: International Sports Analytics Conference and Exhibition. pp. 216–227. Springer (2024)
11. Sepasdar, Z., Gautam, S., Midoglu, C., Riegler, M.A., Halvorsen, P.: Soccergraphrag: Applications of graphrag in soccer. In: International Workshop on Graph-Based Approaches in Information Retrieval. pp. 1–10. Springer (2024)
12. Shin, Y., Park, S., Han, Y., Jeon, B.K., Lee, S., Kang, B.J.: Soccer-clip: Vision language model for soccer action spotting. IEEE Access **13**, 44354–44365 (2025)
13. Strand, A.T., Gautam, S., Midoglu, C., Halvorsen, P.: Soccerrag: Multimodal soccer information retrieval via natural queries. In: 2024 International Conference on Content-Based Multimedia Indexing (CBMI). pp. 1–7. IEEE (2024)
14. Tai, C.Y., Chen, Z., Zhang, T., Deng, X., Sun, H.: Exploring chain-of-thought style prompting for text-to-sql. arXiv preprint arXiv:2305.14215 (2023)
15. The Guardian: Nfl finalizes blockbuster $113bn media rights deal through 2033 season. The Guardian (2021), accessed: 2025-04-17
16. Xia, H., Yang, Z., Zou, J., Tracy, R., Wang, Y., Lu, C., Lai, C., He, Y., Shao, X., Xie, Z., et al.: Sportu: A comprehensive sports understanding benchmark for multimodal large language models. arXiv preprint arXiv:2410.08474 (2024)

17. Xie, X., Xu, G., Zhao, L., Guo, R.: Opensearch-sql: Enhancing text-to-sql with dynamic few-shot and consistency alignment. Proceedings of the ACM on Management of Data **3**(3), 1–24 (2025)
18. Xie, Y., Jin, X., Xie, T., Lin, M., Chen, L., Yu, C., Cheng, L., Zhuo, C., Hu, B., Li, Z.: Decomposition for enhancing attention: Improving llm-based text-to-sql through workflow paradigm. arXiv preprint arXiv:2402.10671 (2024)
19. Yang, Z., Xia, H., Li, J., Chen, Z., Zhu, Z., Shen, W.: Sports intelligence: Assessing the sports understanding capabilities of language models through question answering from text to video. arXiv preprint arXiv:2406.14877 (2024)
20. Yuan, H., Tang, X., Chen, K., Shou, L., Chen, G., Li, H.: Cogsql: A cognitive framework for enhancing large language models in text-to-sql translation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 25778–25786 (2025)
21. Zhang, H., Cao, R., Xu, H., Chen, L., Yu, K.: Coe-sql: In-context learning for multi-turn text-to-sql with chain-of-editions. arXiv preprint arXiv:2405.02712 (2024)
22. Zhang, J., Han, D., Han, S., Li, H., Lam, W.K., Zhang, M.: Chatmatch: Exploring the potential of hybrid vision–language deep learning approach for the intelligent analysis and inference of racket sports. Computer Speech & Language **89**, 101694 (2025)
23. Zhu, Y., Yuan, H., Wang, S., Liu, J., Liu, W., Deng, C., Chen, H., Liu, Z., Dou, Z., Wen, J.R.: Large language models for information retrieval: A survey. arXiv preprint arXiv:2308.07107 (2023)

Appendix

1 Data

1.1 Schema

Table 1. Sample keys from "rushing" schema. The schema contains name of the fields, valid values, types, and corresponding explanations of the fields. The information helps LLMs formulate accurate and executable API calls.

OpenSearch Key	Type	Format	Values	Explanation
nflId	one-of-list	integer		NFL ID is a unique identifier for an NFL player. Rusher ID is the NFL ID of the player who attempted a run on a play.
rushYards	range	float	(0, inf)	Rush Yards is the total distance in yards gained by a rusher on a run play or series of carries.
touchdown	singular	integer	[0, 1]	A rush attempt that results in a touchdown (6 points for the offense).

Table 2. Sample keys from "defense" schema. The schema contains name of the fields, valid values, types, and corresponding explanations of the fields. The information helps LLMs formulate accurate and executable API calls.

OpenSearch Key	Type	Format	Values	Explanation
nflId	one-of-list	integer		NFL ID is a unique identifier of an NFL player. Defender ID is the NFL ID of a player on the field for a defensive scrimmage play.
playerAlignmentEDGE	singular	integer	[0, 1]	The defender is aligned as an edge defender (aka defensive end). alignment = EDGE
alignmentDirection	one-of-list	string	[LEFT, RIGHT]	Alignment Direction is the side of the field the defender lines up on from the perspective of the defense (Left or Right). A player lined up on the left side of the defense is on the offense's right, while a player lined up on the right side of the defense is on the offense's left.

1.2 Question and Answer Pairs

Table 3. Sample queries and their ground truth play counts used for development and evaluation.

Query	Ground Truth Play Count	Complexity	Relevant Schema Fields
How many passes did Patrick Mahomes throw during the 2022 regular season?	648	Easy	Patrick Mahomes 2022 Reg Play Type = Pass
How many touchdown passes greater than 10 yards did Patrick Mahomes throw during the 2022 regular season?	12	Medium	Patrick Mahomes 2022 Reg Play Type = Pass Pass TD — Pass Yards >10
How many touchdown passes greater than 10 yards did Patrick Mahomes throw during the 2022 regular season from Under Center?	1	Difficult	Patrick Mahomes 2022 Reg Play Type = Pass Pass TD — Pass Yards >10 — Team Off Formation = Under Center

2 System Interface

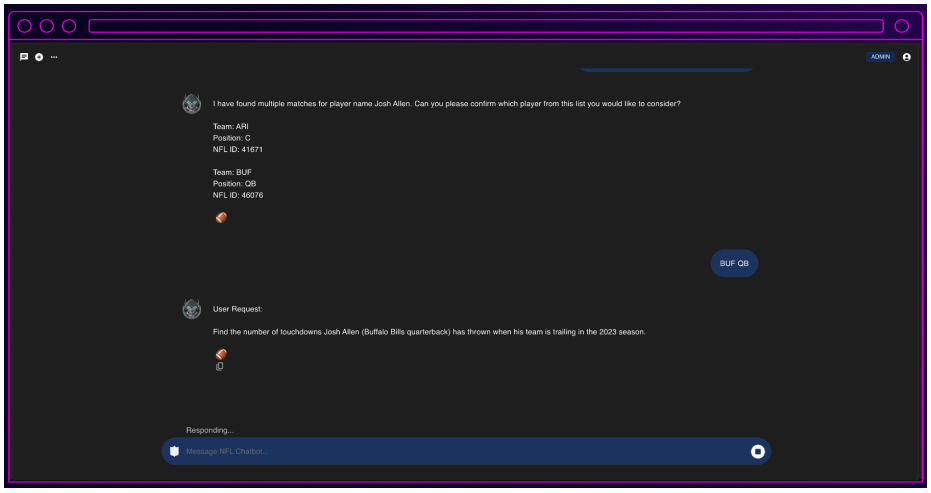

Fig. 1. The solution clarifies with the user when their initial prompt contains a player name that is tied to multiple distinct players in the database.

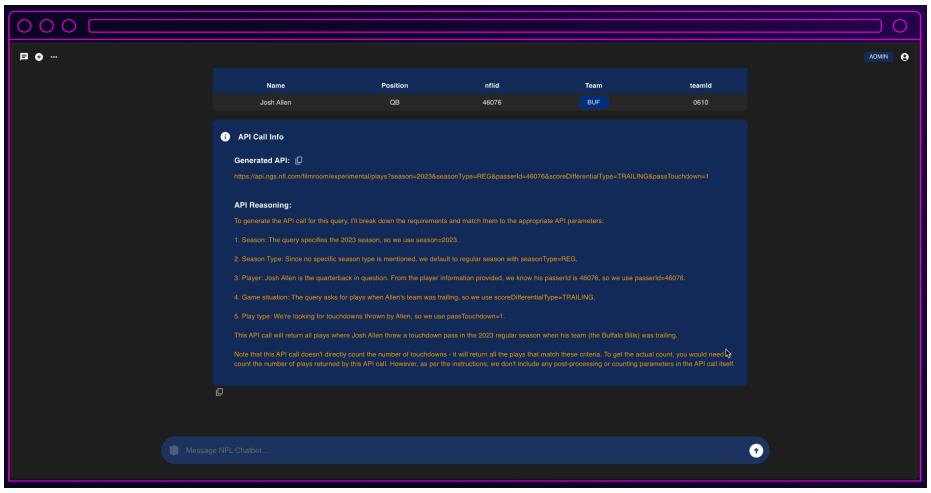

Fig. 2. Reasoning step and formulated API.

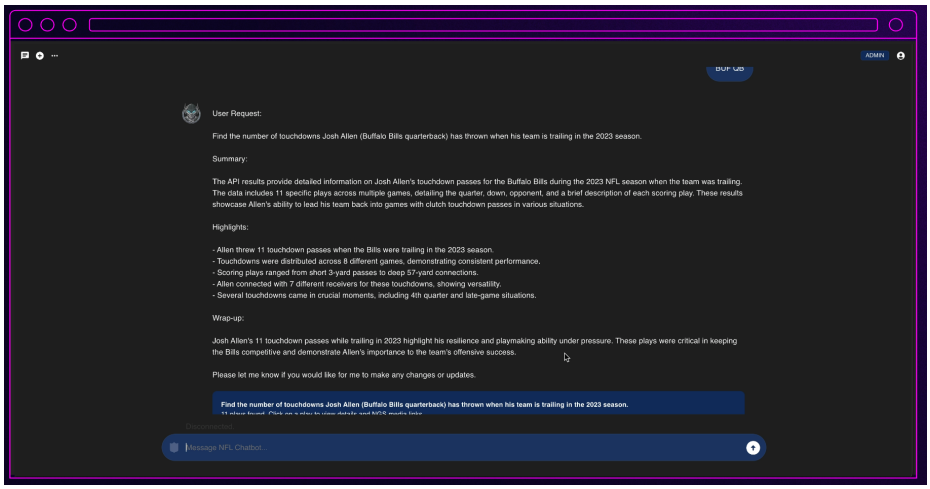

Fig. 3. Natural language response of the query.

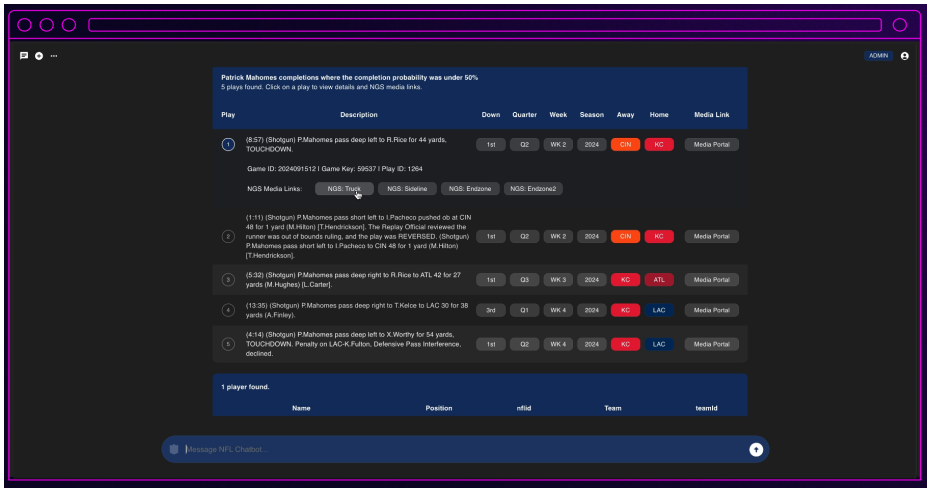

Fig. 4. Links to the retrieved plays' associated media content within a Media Asset Management solution (MAM).

3 System Architecture

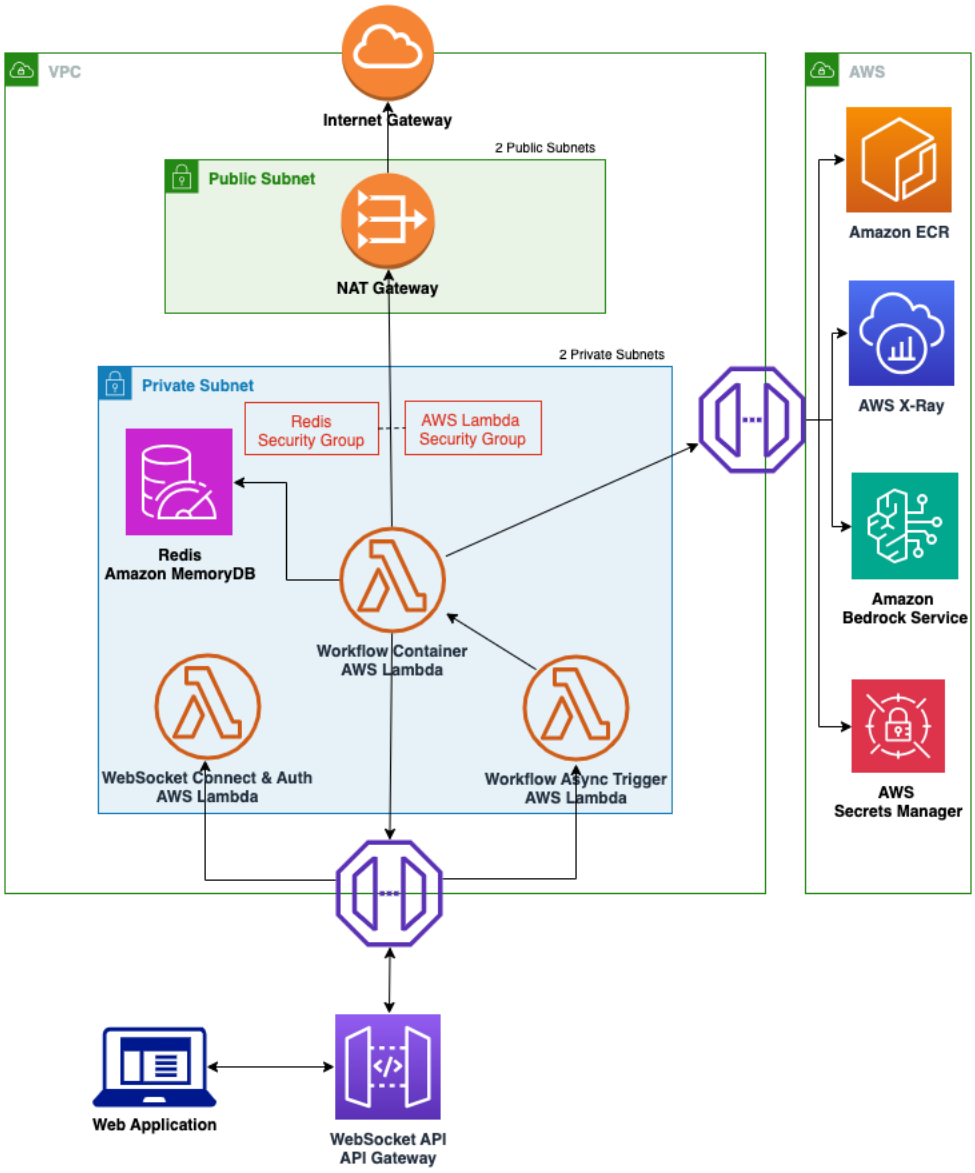

Fig. 5. Overall architecture of the infrastructure in AWS ecosystem.