**🏀 TFM: NBA Flow Agent**

**1. Objetivo general**

Desarrollar un agente conversacional capaz de responder preguntas en lenguaje natural sobre la NBA, integrando procesamiento de lenguaje natural, ejecución de funciones Python y generación de respuestas en lenguaje natural.

**Flujo general:**  
**Usuario → NLU → Router → Executor → Interpreter → Respuesta**

**Ejemplo:**

* **Usuario:** “¿Cuántos puntos promedió LeBron en 2022-23?”
* **Agente:** “LeBron James promedió 28.9 puntos por partido en la temporada 2022-23.”

**Arquitectura general**

* **Frontend (UI web):** donde el usuario escribe preguntas y recibe respuestas.
* **Backend (API):** donde vive la lógica del agente (NLU → Router → Executor → Interpreter).
* **Toolkit NBA:** nba\_agent\_tools.py con las funciones que consultan la API NBA y devuelven DataFrames.

**2. Fases del desarrollo**

**2.1 Definición del toolkit**

* Crear nba\_agent\_tools.py con 15–20 funciones útiles, como:
  + get\_player\_season\_row(player, season)
  + get\_team\_standings(season)
  + get\_player\_gamelog(player, season)
  + get\_league\_leaders(stat, season)
* Objetivo: disponer de un backend robusto que devuelva datos limpios (DataFrame o dict).
* **Entregable:** módulo Python probado con ejemplos.

**2.2 Dataset para entrenamiento del LLM**

* Crear un corpus de ejemplos:
  + Pregunta → JSON con nombre de función + argumentos.
* Ejemplos variados: distintas formas de preguntar lo mismo.
* Guardar en JSONL para usar en el fine-tuning.

**2.3 Fine-tuning / Instrucción del modelo**

* Entrenar con QLoRA un modelo pequeño (Mistral 7B o LLaMA 3 8B).
* Enseñar al modelo a mapear pregunta → JSON.
* Métrica: accuracy en función y argumentos.

**2.4 Integración con el pipeline**

* Conectar el modelo con tu nba\_agent\_tools.
* Validar que hace llamadas correctas y devolver respuestas en lenguaje natural.

**2.5 UI y demo final**

* Montar la UI en Streamlit/Gradio.
* Demostración: el usuario pregunta → el sistema responde con datos reales.

**2.2 Procesamiento del lenguaje natural (NLU)**

* **Intent classification:** identificar la intención del usuario (stats, standings, gamelog, líder de la liga…).
* **Slot filling:** extraer entidades clave:
  + Jugador / equipo
  + Temporada
  + Métrica (puntos, rebotes, asistencias)
  + Tipo de temporada (Regular / Playoffs)
* **Herramientas sugeridas:**
  + spaCy con NER personalizado
  + Expresiones regulares y reglas heurísticas
  + LLM pequeño local para parsing semántico
* **Entregable:** función parse\_user\_input(input\_text) → {intent, slots}

**2.3 Router de intents**

* Mapear (intent + slots) → función concreta del toolkit.
* Ejemplo:
  + Input: "Promedios de Curry en 2015"
  + Output: get\_player\_season\_row("Stephen Curry","2014-15")
* Considerar normalización de nombres de jugadores y equipos.
* **Entregable:** función route\_intent(intent, slots) → función + argumentos.

**2.4 Ejecutor con manejo de errores**

* Llamar a la función con los argumentos limpios.
* Manejar errores:
  + API caídas → retries
  + Datos no encontrados → mensaje claro
* **Entregable:** módulo executor que devuelva DataFrame o dict limpio.

**2.5 Interpreter / generación de respuestas**

* Convertir DataFrame en texto natural.
* Ejemplo:
* Input DF: GP=55, PTS=28.9, REB=8.3, AST=6.8
* Output: "En la temporada 2022-23, LeBron jugó 55 partidos, con promedios de 28.9 puntos, 8.3 rebotes y 6.8 asistencias."
* Estrategias:
  + Plantillas fijas o jinja2
  + LLM para frases más naturales
* **Entregable:** interpret\_result(df, intent, slots) → string final

**2.6 Contexto conversacional**

* Mantener memoria de la sesión para seguimiento de contexto:
  + Ejemplo:
    - Usuario: “Estadísticas de LeBron en 2022-23”
    - Luego: “¿Y en playoffs?”
  + El agente debe inferir el jugador y la temporada.
* Implementar un objeto Session con estado.
* **Entregable:** soporte de contexto simple (en memoria).

**2.7 Iteración y mejoras**

* Comparaciones entre jugadores
* Gráficos automáticos (matplotlib / seaborn)
* Cache de resultados para optimización
* Integración front-end:
  + Dashboard rápido: Streamlit
  + API REST: FastAPI
* Posible integración de LLM para mejorar fluidez y redacción natural

**3. Stack recomendado (mínimo viable)**

* **Python core:** nba\_api, pandas
* **NLU:** spaCy + reglas, o LLM local
* **Infraestructura:** FastAPI / Streamlit
* **Interpretación:** plantillas + heurísticas → LLM

**4. Primer milestone (MVP)**

* Script de consola:
  1. Pregunta en lenguaje natural
  2. Detecta intent y entidades (regex/simple NLU)
  3. Llama a nba\_agent\_tools
  4. Devuelve frase interpretando el DataFrame
* **Objetivo:** probar flujo completo de principio a fin con un ejemplo real, sin interfaz web ni gráficos.

**Diagrama de flujo del Agente Conversacional NBA**

|  |  |  |  |
| --- | --- | --- | --- |
| **Etapa** | **Entrada** | **Proceso** | **Salida / Ejemplo** |
| **1. Usuario** | Pregunta en lenguaje natural | — | “¿Cuántos puntos promedió LeBron en 2022-23?” |
| **2. NLU (parse\_user\_input)** | Texto del usuario | Clasifica intención + extrae entidades (jugador, temporada, métrica, tipo de temporada) | {'intent': 'player\_stats', 'player': 'LeBron James', 'season': '2022-23', 'metric': 'PTS'} |
| **3. Router (route\_intent)** | Intent + slots | Selecciona función del toolkit y prepara argumentos | get\_player\_season\_row("LeBron James","2022-23") |
| **4. Executor** | Función + argumentos | Llama a nba\_agent\_tools, maneja errores | DataFrame: GP=55, PTS=28.9, REB=8.3, AST=6.8 |
| **5. Interpreter (interpret\_result)** | DataFrame + intent + slots | Convierte datos en lenguaje natural, usando plantillas o LLM | “En la temporada 2022-23, LeBron jugó 55 partidos, con promedios de 28.9 puntos, 8.3 rebotes y 6.8 asistencias.” |
| **6. Output** | Texto generado | — | Mensaje al usuario |

|  |  |  |  |
| --- | --- | --- | --- |
| **Módulo** | **Responsabilidad** | **Herramientas / Notas** | **Entregable** |
| **NLU** | Detectar intención y extraer entidades | spaCy, regex, heurísticas, LLM local | parse\_user\_input() |
| **Router** | Mapear intent + entidades → función concreta | Funciones del toolkit, normalización de nombres | route\_intent() |
| **Executor** | Ejecutar función y manejar errores | pandas, nba\_api, retries | DataFrame limpio o dict |
| **Interpreter** | Convertir datos en texto natural | Plantillas (jinja2) o LLM | interpret\_result() |
| **Contexto** | Mantener memoria de la sesión | Objeto Session, estado en memoria | Soporte de contexto multi-turno |
| **Frontend / Output** | Mostrar respuesta al usuario | Consola, Streamlit, FastAPI | Texto interpretado |

**Responsabilidades y herramientas de cada módulo**

**Funciones / preguntas que el agente debe poder responder**

**🔹 Estadísticas de jugadores**

1. Mostrar los promedios y totales de un jugador en una temporada específica, incluyendo la opción de por 36 minutos o en playoffs.
2. Consultar los promedios por partido de un jugador en una temporada concreta.
3. Identificar el partido en el que un jugador anotó más puntos en su carrera o en una temporada específica.
4. Consultar los últimos N partidos de un jugador, con opción de filtrar solo playoffs.
5. Mostrar la evolución de un estadístico de un jugador a lo largo de una temporada (por ejemplo, sus puntos por partido semana a semana).
6. Consultar el total de dobles y triples dobles de un jugador, ya sea en toda su carrera, en una temporada o en playoffs.
7. Consultar los récords personales de un jugador, como su mayor cantidad de puntos, rebotes o asistencias en un partido.
8. Consultar los premios, logros y reconocimientos obtenidos por un jugador a lo largo de su carrera.
9. Mostrar todos los jugadores seleccionados para el **All-NBA Team** en una temporada concreta.
10. Consultar todos los jugadores que participaron en el **All-Star Game** de una temporada determinada.

**🔹 Comparaciones y rankings**

1. Saber qué jugador lideró la liga en puntos, rebotes, asistencias u otra estadística en una temporada determinada.
2. Mostrar los N mejores jugadores de la liga en una estadística determinada, por temporada o playoffs.
3. Contar cuántos jugadores promediaron un valor igual o superior a cierta cifra en una temporada (por ejemplo, puntos por partido).
4. Contar cuántos jugadores cumplieron varios criterios al mismo tiempo, como puntos, rebotes y asistencias, en una temporada determinada.
5. Comparar un estadístico entre dos jugadores, ya sea en una temporada concreta o a lo largo de su carrera.
6. Comparar estadísticas de carrera entre varios jugadores (por ejemplo, quién tiene más puntos totales).
7. Contar cuántos jugadores superaron un determinado total en su carrera en cualquier estadística.
8. Consultar los **premios individuales de la liga** de una temporada (MVP, Rookie del Año, DPOY, etc.).
9. Consultar los máximos anotadores o líderes por posición dentro de un equipo en una temporada determinada.
10. Comparar el desempeño de un equipo contra otro a lo largo de la temporada o en playoffs, incluyendo resultados y estadísticas promedio.

**🔹 Partidos**

1. Mostrar cómo fue un partido entre dos equipos en un año específico, incluyendo resultados y estadísticas.
2. Obtener el **boxscore de un partido** completo, con estadísticas individuales de todos los jugadores.
3. Mostrar los últimos N partidos de un equipo.
4. Consultar el desempeño de un equipo en todos sus partidos de playoffs en una temporada determinada.
5. Consultar enfrentamientos directos (head-to-head) entre dos jugadores o equipos en una temporada o a lo largo de su carrera.

**🔹 Equipos y temporadas**

1. Mostrar la tabla de posiciones de la NBA en una temporada, por conferencia o global, incluyendo playoffs si se desea.
2. Consultar cómo le fue a un equipo durante toda una temporada, incluyendo victorias, derrotas y estadísticas generales.
3. Obtener la plantilla completa de un equipo en una temporada determinada (titulares, suplentes, rotación).
4. Generar resúmenes estadísticos por equipo: puntos por partido, rebotes, asistencias, porcentaje de tiros, etc., para toda una temporada o playoffs.
5. Consultar el ranking de equipos por una métrica específica en una temporada (ejemplo: mejor ofensiva, mejor defensa).