Industry: Sports
Challenge 1: Deepening Fan Engagement
Team: SJSU Dream Team
Agent Name: *FanEngage*
Members: Tyler Co Seng

Problem Statement
A new ownership group has bought and taken over a sports team and wants to modernize how it engages with its fans. Currently, fans interact with the team across social media, apps, notifications, streaming, etc, which constantly creates a flow of reactions and expectations. However, most communication can still be improved and is not responsive to what is actually happening in the game or what specific fans care about.

The Sports Industry Challenge 1 states: “How might we use AI agents to automatically understand fan sentiment, deliver personalized content, and streamline interactions to create a more delightful and efficient fan experience?”. This means I need to find a way to react to key game moments in real time, follow fan preferences (favorite players and spoiler settings), and deliver short messages that can be pushed across channels (in-app, push notifications) without copywriting every time.

Solution Overview - FanEngage AI Agent
Our solution to this is *FanEngage*, an AI agent that generates personalized, spoiler-free fan messages for different game moments using IBM watsonx.ai and the Granite 3-8B Instruct model. Instead of manually writing messages for every scenario, *FanEngage* accepts a structured JSON event that takes:

- `event_type`: “`pre_game`”, “`in_game_close`”, “`post_game`”, “`big_play`”, “`milestone`”
- `team`, `opponent`: which teams are playing
- `outcome`: “`win`”, “`loss`”, or “`n/a`” for live events
- `spoiler_setting`: “on” or “`off`” to control whether it's spoiler-free

With these inputs, the agent calls the Granite-3B prompt that codes rules for tone, personalization, spoiler setting, and channel behaviour. Then it returns a JSON response with:

- `message`: short 1-2 sentences
- `tone`: “`empathetic`”, “`celebratory`”, “`excited`”
- `channel`: taken from input to align with the delivery channel
- `cta`: call to action like “`view_recap`”, “`tune_in`”, “`view_highlights`”

Right now, the prototype focuses on the team Golden State Warriors and the player Stephen Curry, but it is designed so that the same event schema and prompt can be extended across teams and players. Since I did not have Orchestrate access in my environment, I implemented *FanEngage* as a notebook-based agent that calls the watsonx.ai text generation API directly from Python, which follows the program’s instructions to build a watsonx-based prototype.

Key Features and Scenarios

- Post-game loss
    - Empathetic tone, does not push merch, highlights key moments
    - Example: “Tough one vs the Kings. Here are a few key moments you can revisit without spoilers.”
- Post-game win
    - Celebratory tone, focuses on big plays and best moments
    - Example: “What a game! Steph was on fire. Check out his top plays from the win.”
- Pre-game hype
    - Builds hype before tip-off, personalized to follow players and avoid spoilers
- In-game: Close game
    - High-priority messaging when the score is tight, and encourages tuning in live without showing the final result
- In-game: Big play moment
    - Triggered by big plays or stat spikes (30-point game)
- Milestone moment
    - Celebrates milestones (career points, highest scoring game, etc.) for followed players and directs fans to highlight reels

Across all the scenarios, the agent respects spoiler settings when `spoiler_setting = “off”`, and the agent avoids the final scores or saying who won/lost. It also personalizes content, like if the fan follows Stephen Curry, the message is centered around that player when it's appropriate.

Alignment with the Challenge Requirements

This solution is designed to address the Challenge 1 solution requirements:

1. Sentiment Analysis and Response Generation
    - The current prototype approximates sentiment awareness through event context and rules stated in the prompt. For example, empathetic for losses, celebratory for wins, and excited/urgent for close games.
    - Granite 3-8B generates a natural language response that reflects this tone and respects the spoiler setting.
2. Personalized Content
    - This happens through the “`follows`” field. If a fan follows a certain player, the agent mentions that player by name and tailors the description to their performance.
    - Different events trigger different messages (pre-game hype vs. milestone celebration), which improves how relevant the moment is.
3. Smooth Experience Across Platforms
    - The agent includes a “`channel`” in both the input and output (“`in_app`”, “`push`”) and selects CTAs that will make sense for that specific channel.
    - The prototype runs in a notebook rather than a production application. The JSON in/out construct is designed so that a notification system can be integrated with it.
4. Scalable and Secure Data Handling

    - The prototype uses a small, structured JSON event schema, which can be extended without changing the core prompt.
    - It uses artificial fan and game events and avoids any personal information. This is to stay consistent with the program guidelines.

Next Steps

- Re-implementing the agent in watsonx Orchestrate once I gain access, so the workflows can be configured visually.
- Building a simple front-end or notification service that calls the notebook/agent API and delivers messages to real users.

Link to Prototype

https://github.com/TylerCoSeng/fanengage-sports-ai-lab/tree/main