Students: *Johan Balceiro*
Mentor: *Gummi Oxndal*
Product Owner: *Gummi Oxndal*
Instructor: *Instructor/Faculty: Masoud Sadjadi, FIU*

Florida International University

Fall 2023 Capstone II Project

SkillCourt

Introducing SkillCourt: The Ultimate Soccer Training App!

SkillCourt is a cutting-edge application designed to revolutionize the way soccer athletes hone their skills. This innovative platform provides a controlled environment for players to engage in dynamic drills, personalize training regimens, and collaborate seamlessly with coaches. The app communicates with interactive pads that respond to hits and light up with LED feedback, offering real-time insights into performance.

Key Features:

- Elevate Your Skills: Engage in dynamic drills for precision and mastery.
- Personalized Regimens: Craft customized routines targeting specific plays and habits.
- Coach Collaboration: Seamlessly interact with coaches for expert guidance.
- Interactive Feedback: LED-enhanced pads provide real-time insights.
- Controlled Environment: Practice in a dynamic yet controlled space.

SkillCourt – Precision, Collaboration, Innovation, Excellence. Elevate your soccer game to new heights!

Server Side (Pad) Implementation

In the server-side implementation for the interactive pads, SkillCourt leverages the Python programming language. The server is responsible for handling communication with the application, processing hits, and triggering LED feedback. Here's a concise overview of the implementation:

- WebSocket Integration: Utilized Python's WebSocket libraries, such as websockets or socket.io, to establish a bidirectional communication channel with the Flutter application.
- Event Handling: Implemented event-driven mechanisms to handle incoming hit data from the application. This involves processing hit patterns and triggering corresponding LED responses.
- LED Control Logic: Developed logic to control the LED illumination patterns on the interactive pads based on real-time data received through the WebSocket connection.
- Concurrency and Threading: Implemented concurrency mechanisms, such as threading, to manage simultaneous communication with multiple pads, ensuring efficient and responsive handling of hits.
- Error Handling: Implemented robust error-handling mechanisms to gracefully manage communication disruptions or unexpected events, ensuring the stability of the server-side logic.

Client Side (App) Implementation

WebSocket Integration: Leveraged Flutter packages like web_socket_channel to establish and manage WebSocket connections with the Python server.

- User Interface (UI): Designed an intuitive and user-friendly interface to enable athletes to interact with the application effortlessly. This includes features for selecting drills, creating personalized regimens, and initiating training sessions.
- Data Serialization: Implemented mechanisms for serializing and deserializing data to be exchanged between the application and the server, ensuring a standardized communication protocol.
- Real-Time Feedback Display: Integrated components to visually represent real-time feedback from the interactive pads, allowing users to track their performance during training sessions.
- Error Handling and Resilience: Incorporated error-handling strategies to gracefully manage communication disruptions or errors, providing a resilient user experience.
- Cross-Platform Compatibility: Ensured compatibility with multiple platforms (iOS and Android) by utilizing Flutter's inherent cross-platform capabilities

Communication Evolution

SkillCourt initially employed the mDNS (Multicast Domain Name System) as its communication protocol with the interactive pads. However, in our pursuit of providing users with an even more seamless and robust experience, we have made a strategic shift to the WebSocket protocol.

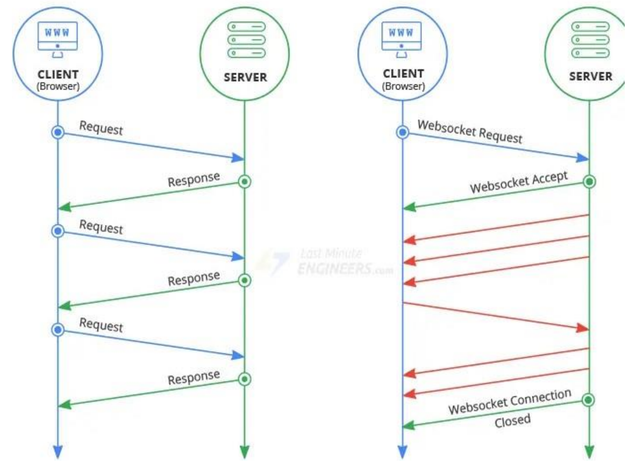
Why the Change?

Enhanced Real-Time Responsiveness: WebSocket offers a more efficient and low-latency communication channel compared to mDNS, ensuring that the feedback from the interactive pads is even more instantaneous.

Reliability and Stability: WebSocket's persistent connection architecture enhances reliability, reducing the chances of communication disruptions during training sessions. This change ensures a more stable and consistent user experience.

Scalability: WebSocket's support for bidirectional communication makes it inherently scalable, allowing SkillCourt to accommodate future enhancements and features seamlessly.

Cross-Platform Compatibility: WebSocket is widely supported across various platforms and devices, ensuring a consistent and uniform experience for SkillCourt users, regardless of their chosen device.



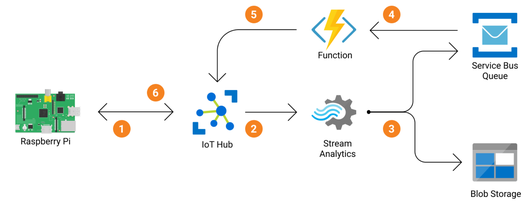
Summary

As a pivotal member of the SkillCourt project, my primary role revolved around orchestrating the transition between communication protocols, a critical evolution that significantly impacted the app's performance. I actively contributed to the initial research and decision-making process that led to the strategic migration from mDNS to the WebSocket protocol.

This involved delving into the intricacies of each protocol, analyzing their strengths and weaknesses, and ultimately steering the team towards the adoption of WebSocket for its superior bidirectional communication capabilities.

In addition to this, I took the initiative to create a robust and testable emulator in IoT Hub Azure. This emulator serves as a valuable asset for future teams, enabling them to simulate real-world scenarios and thoroughly test the communication system without the need for physical pads. The emulator not only streamlines the development process but also enhances the overall reliability and scalability of SkillCourt.

Being at the forefront of these technical decisions and implementations has been an enriching experience, and I'm proud to have played a key role in enhancing the communication infrastructure of SkillCourt for a more responsive and innovative soccer training platform



References

Mozilla Developer Network. (n.d.). Writing WebSocket servers. Retrieved from https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API/Writing_WebSocket_servers

Microsoft. (n.d.). Understand Different Connection Strings in Azure IoT Hub. Retrieved from https://devblogs.microsoft.com/iotdev/understand-different-connection-strings-in-azure-iot-hub/

Microsoft. (n.d.). IoT Hub live data visualization in web apps. Retrieved from https://learn.microsoft.com/en-us/azure/iot-hub/iot-hub-live-data-visualization-in-web-apps

REQUIREMENTS

Flutter
Dart
GitHub
Android Studio
Android VM
Python
PyCharm
Android Tool Chain

FIU Engineering & Computing