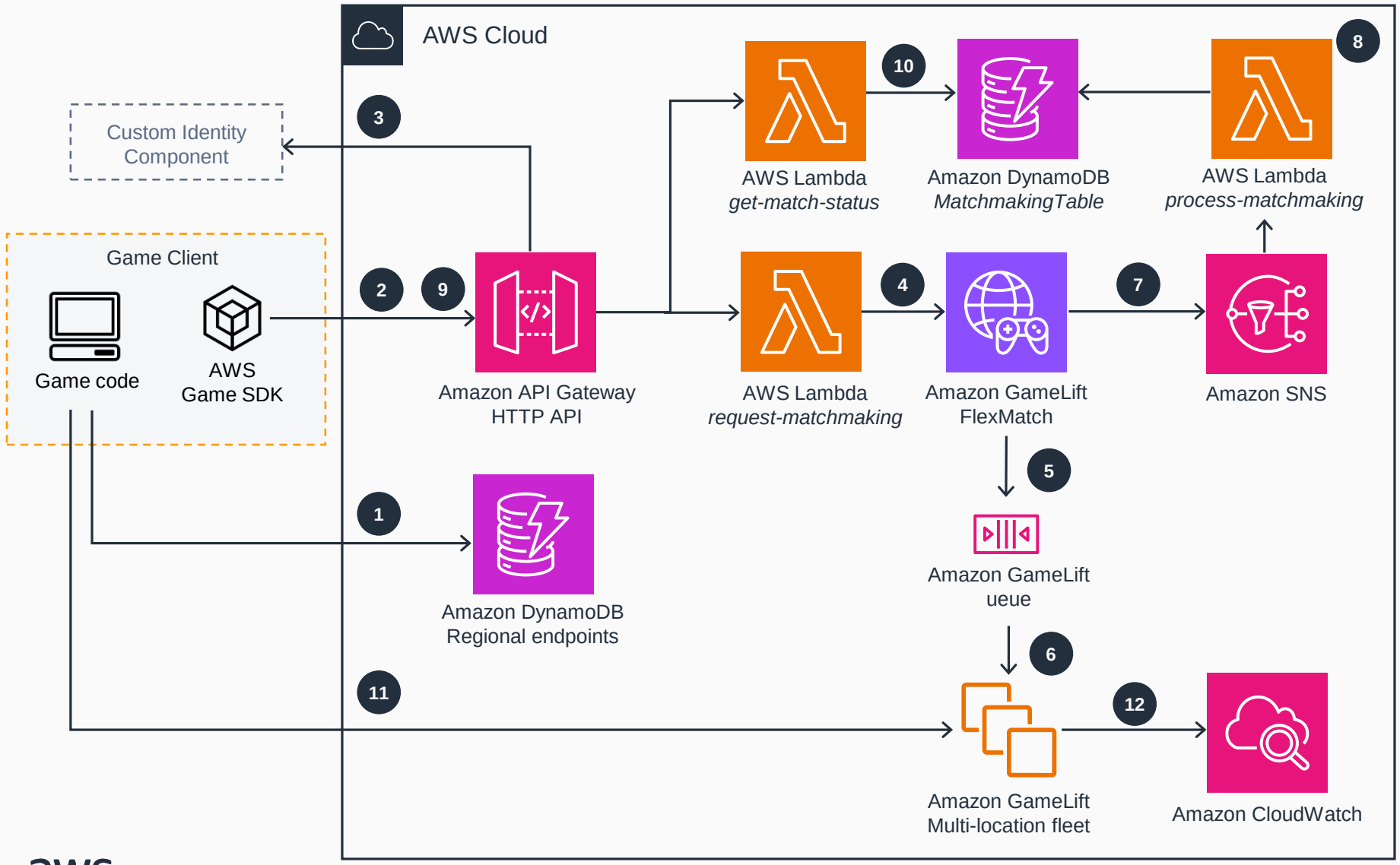
Guidance for Multiplayer Session-Based Game Hosting on AWS

This architecture diagram shows how to build a global multiplayer game on Amazon GameLift with matchmaking and a serverless backend.



1. Game client measures TCP latency to AWS Regions by calling **Amazon DynamoDB** endpoints.
2. Game client uses the AWS Game SDK to make an authenticated POST request to **Amazon API Gateway** with the latency data in the request body.
3. **API Gateway** validates client JSON Web Token with the Custom Identity Component public keys.
4. **API Gateway** calls *request-matchmaking* **AWS Lambda** function, which sends a *StartMatchmaking* request to **Amazon GameLift FlexMatch** with the latency data.
5. **Amazon GameLift FlexMatch** matches the player with other players and calls the **Amazon GameLift** queue to request a placement in case of a new match. It can also backfill players to existing matches.
6. The **Amazon GameLift** queue finds a placement based on player latencies in one of the **Amazon GameLift** fleet locations.
7. Once the placement is done and session started, **Amazon GameLift FlexMatch** sends a *MatchmakingSucceeded* event to an **Amazon Simple Notification Service (Amazon SNS)** topic. It also sends all intermediate events such as *MatchmakingSearching*.
8. **Amazon SNS** invokes *process-matchmaking* **Lambda** function, which updates all match status changes to a **DynamoDB** table.
9. Game client polls match status with a GET request containing the matchmaking ticket ID.
10. *Get-match-status* **Lambda** function gets the latest match info from **DynamoDB** and sends it back to the game client. When matchmaking is done, it also sends the IP, port, and player session ID to the client.
11. Game client connects with TCP (often UDP in real-time games) to the game session and sends the player session ID that the game server validates.
12. The instances send logs and metrics to **Amazon CloudWatch** using the **CloudWatch** agent.

Reviewed for technical accuracy February 14, 2024

AWS Reference Architecture