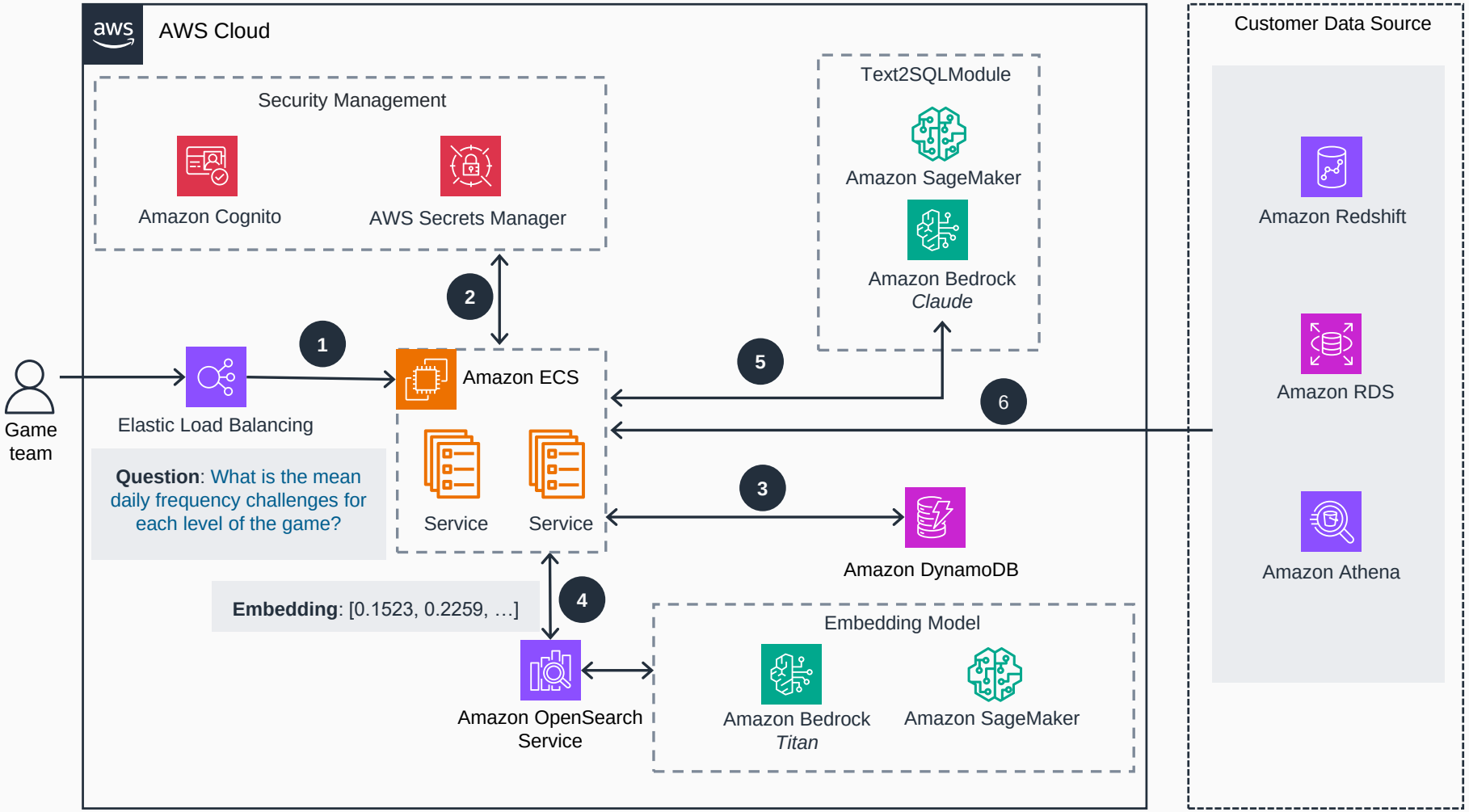
Guidance for Creating Player Insights with Generative AI on AWS

This architecture diagram demonstrates how game studios can unlock insights and respond to player behavior using natural language and generative business intelligence. Powered by LLMs in Amazon Bedrock and enhanced using RAG, this Guidance shows you how to execute SQL queries that answer business-related questions and create dashboards for player analysis. This results in timely player insights, quicker business decisions, and a reduced burden on game analytics teams.



1. The game operations team start a search request and query from a frontend application hosted on **Amazon Elastic Container Service (Amazon ECS)**.
2. **Amazon Cognito** handles user authentication and authorization, helping ensure secure access to the services. **AWS Secrets Manager** securely stores and retrieves sensitive information, such as database credentials, used by the services.
3. **Amazon DynamoDB** stores user profiles and related data, providing a scalable and high-performance database.
4. The embedding module leverages **Amazon OpenSearch Service** and embedding models from **Amazon SageMaker** to process and index for efficient querying.
5. An LLM hosted on **Amazon Bedrock** or **SageMaker** converts natural language text into SQL queries.
6. The system pulls data definition language (DDL) information and SQL query results from customer data sources such as **Amazon Redshift, Amazon Relational Database Service (Amazon RDS), Amazon Athena** and third-party data sources.

Reviewed for technical accuracy October 23, 2024

AWS Reference Architecture