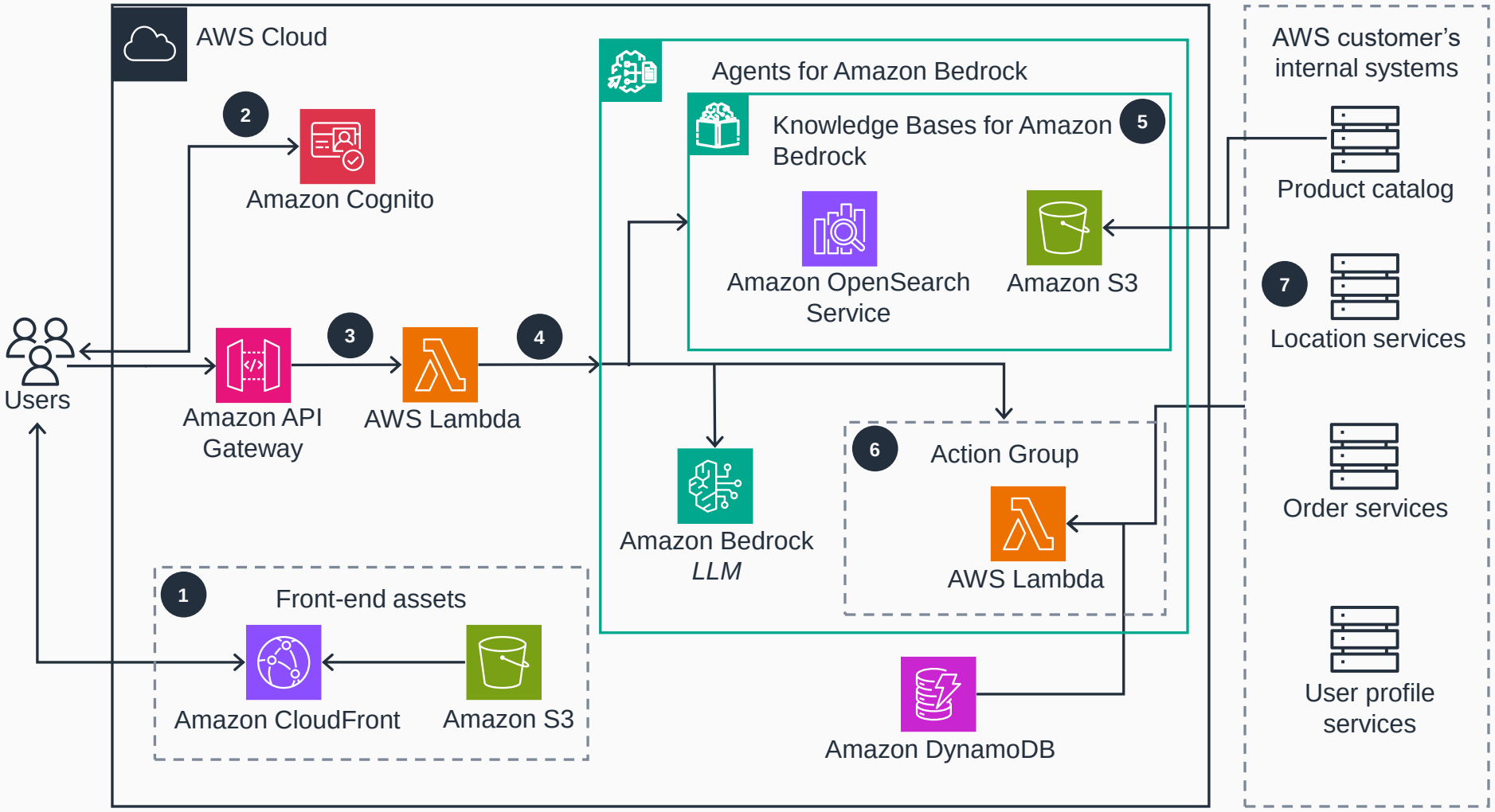
Guidance for Building Custom Chatbots for Order Recommendations Using Agents for Amazon Bedrock

This architecture diagram shows how to build a serverless, scalable generative AI chatbot using both Agents for Amazon Bedrock and Knowledge Bases for Amazon Bedrock. The chatbot can integrate with internal systems to provide personalized recommendations, order placement, and order status.



1. Users access the webpage, which is served by **Amazon CloudFront** and backed by **Amazon Simple Storage Service** (Amazon S3) for website and configuration file storage. Customers can request recommendations or place orders directly from the webpage.
2. The user authenticates with **Amazon Cognito**.
3. **AWS Lambda** uses **Amazon API Gateway** to handle user requests for recommendations and order placement.
4. **Lambda** uses the InvokeAgent API to initiate calls to Agents for **Amazon Bedrock.** Agents automate prompt engineering, invoke large language models (LLMs), and orchestrate the user-requested task by dynamically invoking APIs. **Amazon Bedrock** offers access to foundation models to build generative AI applications.
5. Agents for **Amazon Bedrock** query the Knowledge Bases for **Amazon Bedrock** Retrieve APIs to retrieve relevant text, such as the product catalog from **Amazon OpenSearch Service**, and augment prompts during inference. The Knowledge Bases for **Amazon Bedrock** are set up with **Amazon S3** and **OpenSearch Service** to automate the end-to-end Retrieval Augmented Generation (RAG) workflow.
6. Agents for **Amazon Bedrock** invoke the **Lambda** function from their action group, along with function detail parameters or Open API schema configuration, to execute actions such as placing orders, searching for nearest locations, and retrieving customer order history.
7. **Lambda** integrates with the user's internal systems and **Amazon DynamoDB** to provide the Agents with the appropriate data.

AWS Reference Architecture