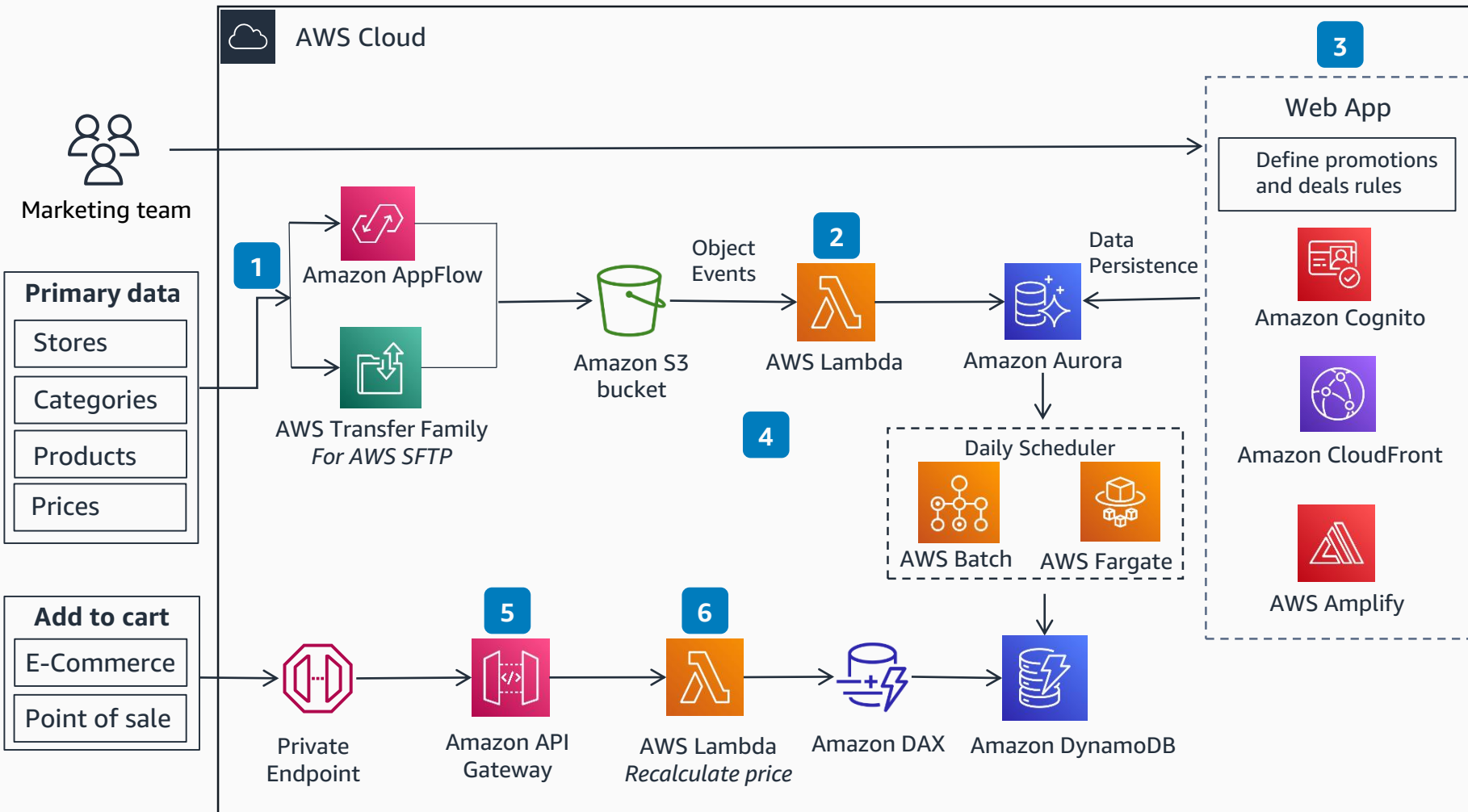


Guidance for Price and Promotion Engine on AWS

The pricing engine in this architecture prices the contents of an online or in-store shopping cart based on set rules, allowing for complex promotions, deals, and dynamic pricing.



1 The pricing engine relies on primary data from systems such as SAP, Product Information Management System (PIMS), or Master Data Management (MDM). **Amazon AppFlow** is used for turn-key integrations with systems such as SAP and systems that are limited to file-based integrations. **AWS Transfer Family** manages secure file transfers, such as secure file transfer protocol (SFTP) jobs.

2 Primary data from **Amazon Simple Storage Service (Amazon S3)** is batch-ingested into a relational database management system (RDBMS) and hosted in **Amazon Aurora** using **AWS Lambda**.

3 The Marketing team uses Web App to create promotions and deals that are stored in **Aurora**. Web App is a scalable, secure, and serverless front-end application, created with **AWS Amplify** for easy code development, **Amazon Cognito** for identity management, and **Amazon CloudFront** for content distribution.

4 **AWS Batch** and **AWS Fargate** serverless runtime processes complex promotions, deals, and dynamic pricing data at a given scheduled time, such as the end of the day. These services store eligible records in **Amazon DynamoDB**.

5 A customer facing commerce system, such as e-Commerce or point of sale, requests the Price Engine API through an **Amazon API Gateway** private endpoint on every "Add to cart" event to re-calculate the price based on configured promotions and deals.

6 **Lambda** calculates the price of respective products from the cart, based on rule records defined in **DynamoDB**. **Amazon DynamoDB Accelerator (DAX)** in-memory cache reduces read latency from **DynamoDB**, which also helps reduce overall API response time to less than a second.