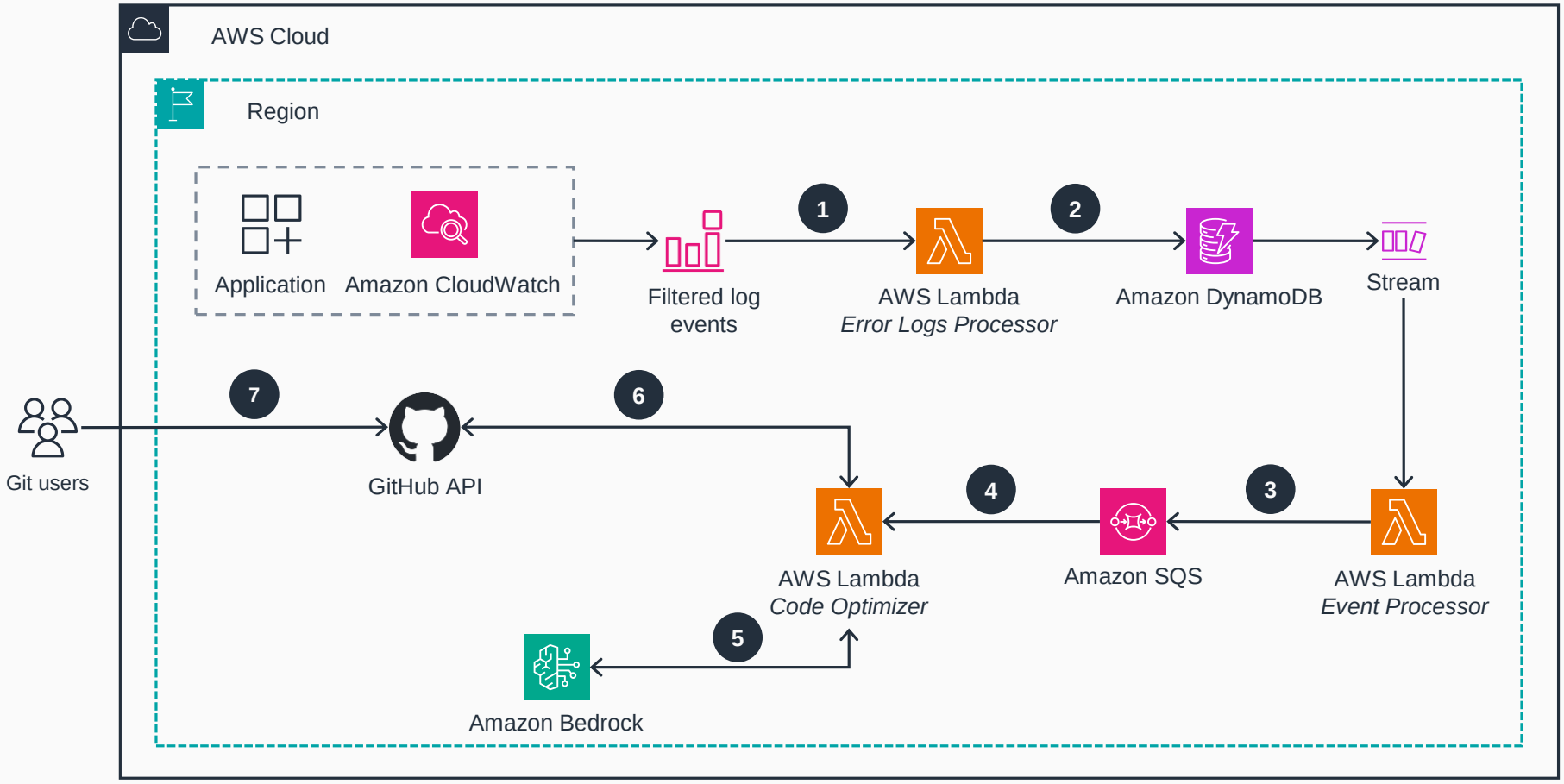
Guidance for Self-Healing Code on AWS

This architecture diagram shows a self-healing code system which ingests an application's error logs and source code to produce modified application code that addresses bugs.



1. The **AWS Lambda** *Error Logs Processor* function receives application error logs through an **Amazon CloudWatch** logs subscription and filter, which matches only Python stack traces. All **Lambda** functions assume an **AWS Identity and Access Management (IAM)** role scoped with minimum permissions to access the required resources.
2. The stack trace in the application error log is md5-hashed for uniqueness and stored in an **Amazon DynamoDB** table to track its processing state. Each item in the table represents a deduplicated error message.
3. The **Lambda** *Event Processor* function obtains events from the **DynamoDB** stream and sends them to **Amazon Simple Queue Service (Amazon SQS)** for batch processing.
4. **Amazon SQS** enqueues messages to enable batch processing and concurrency control for the **Lambda** *Code Optimizer* function.
5. The **Lambda** *Code Optimizer* function builds a prompt that includes source code and the relevant error message. The SSH key to access the Git repository is retrieved from **Parameter Store**, a capability of **AWS Systems Manager**. It invokes **Amazon Bedrock** with the prompt, which returns modified source code as a response. **Amazon Bedrock** is a fully managed service that offers a choice of high-performing foundation models (FMs) through a single API, along with a broad set of capabilities you need to build generative AI applications.
6. The **Lambda** *Code Optimizer* function commits the modified source code into a new Git branch. The Git branch and its corresponding pull request are pushed to the source control system through the GitHub API.
7. Git users review the pull request for testing and integration.

Reviewed for technical accuracy March 18, 2024

AWS Reference Architecture