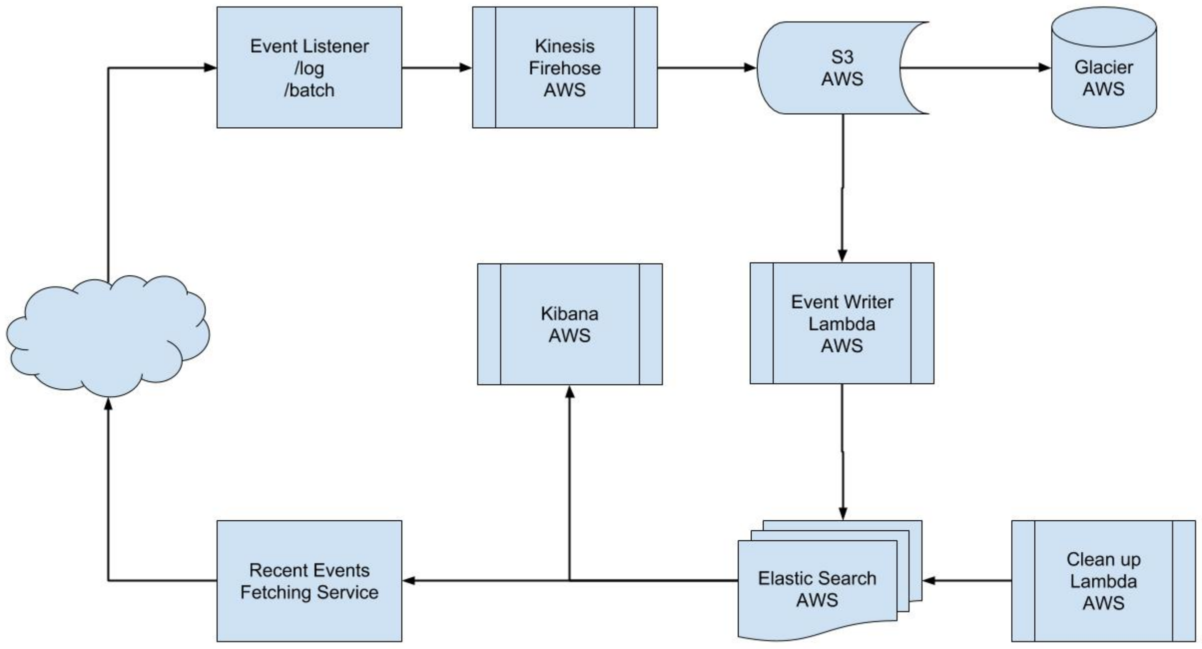
Challenge 2

The diagram below shows the overall architecture of the required system. This architecture makes the assumption that the user events are stateless and eventual consistency in persisting these events is acceptable.



A few notes to supplement this architecture:

- A thin event listener client provides endpoints for real-time and batch processing of events. The listener throws these events to Kinesis Firehose service in AWS.
- All events processed by Firehose are stored in S3 for a scalable, distributed storage.
- All objects in S3 have a lifecycle configuration of transiting to Glacier after 1 month. This is because 4000 events per second each of which is 1 KB equate to roughly 1TB of data over the period of a month.
- A Lambda transferring most recent events from S3 to Elasticsearch triggers every minute and transfers recent events of a user.
- Elasticsearch indices the events by timestamp and user ID.
- A Lambda cleaning up Elasticsearch can trigger every five minutes and drops all events older than five minutes.
- Data from Elasticsearch can be visualized in Kibana.
- A service also interfaces with Elasticsearch to provide users last 100 events.