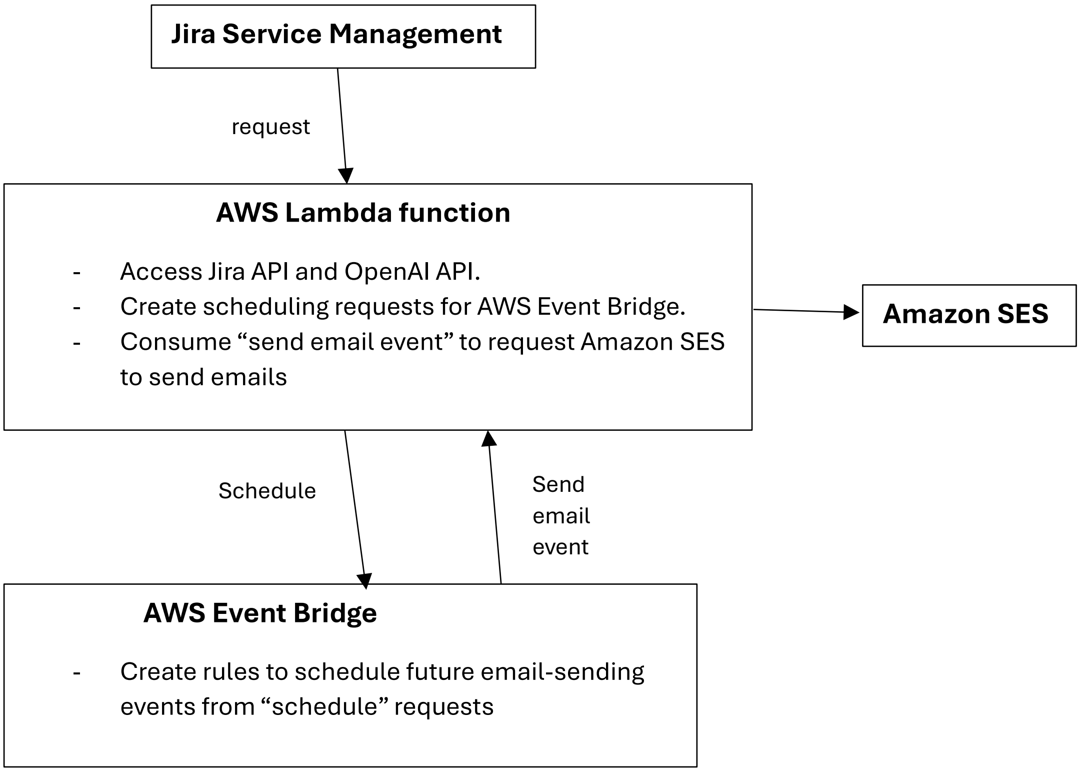
Question 3

The architecture of the email-sending automation project is outlined below.

First, Jira Service Management's rules are configured, so that whenever a release creation or update action is performed by the team on Jira, a request with all the release information is sent to an AWS Lambda Function.

The AWS Lambda Function will be responsible for getting extra information from Jira API using the request payload to compose and send the appropriate scheduling requests containing the project id, release id, the specific points in time that emails need to be sent out and the client's information to AWS Event Bridge.

The AWS Event Bridge will then create or update rules to schedule future requests which will be sent to the Lambda function to send the emails.



Jira Service Management

request

AWS Lambda function

- Access Jira API and OpenAI API.
- Create scheduling requests for AWS Event Bridge.
- Consume "send email event" to request Amazon SES to send emails

Amazon SES

Schedule

Send email event

AWS Event Bridge

- Create rules to schedule future email-sending events from "schedule" requests

At the time of sending the emails, the AWS Event Bridge will send a request to our Lambda server with all the previous information to trigger the process. The Lambda server in turn will access the OpenAI API and Jira API to get the mail content, compose the emails and send the emails to the clients using Amazon SES.

Technology Justification

- **Jira Service Management:** This technology is built into Jira Cloud. It creates an event for every action performed on Jira and supports rules that trigger in response to the event. However, it lacks REST API support for scheduling and does not support actions to create schedules or run scripts in response to events. This is where AWS Event Bridge becomes valuable.

- **AWS Event Bridge:** By leveraging AWS Event Bridge's REST API support, we can automatically schedule an email-sending event upon receiving a Jira event.
- **AWS Lambda Function:** This helps us to run scripts, access APIs and transform inputs. Considering the number of release updates and creations, the server may remain idle for extended periods. Consequently, using a server that operates continuously and charges by the hour can be inefficient. A serverless model is ideal in this scenario, as it allows us to pay only for the resources consumed.

Email Sending:

- Amazon SES API is used to send emails. All communications are encrypted using TLS through the Amazon SES HTTPS endpoint, which supports TLS 1.2 and TLS 1.3.
- The information about the recipients of each release is taken from the database and stored in each request from AWS Event Bridge.
- For email authentication, Amazon SES supports DMARC together with SPF and DKIM to increase trust in the domain and delivery of the emails.
- The scheduling of emails is ensured by the AWS Event Bridge.

Error Handling:

- Our email-sending system will ensure that failed requests are automatically retried at progressively increasing intervals. If a request continues to fail after multiple attempts, it will be rescheduled for future processing. In cases where a request fails after several rescheduling attempts, it will be permanently removed from the queue to maintain system efficiency.
- We can also deploy cross-region servers to improve the availability of the Lambda Function.

Testing and Monitoring:

- Most of our logic is in the Lambda Function so most of the testing will need to be done there. Unit testing is the most suitable in this case because we have clear-cut expectations for each functionality.
- Amazon SES provides a dashboard that helps monitor daily email usage and sending statistics. These numbers can be used to cross-check with our data to detect unusual behaviors.
- Observability can be further improved by implementing OpenTelemetry into our Lambda Function and collecting the exact traces and metrics needed to debug the application such as the time, content and number of the emails sent out for each project or release.

Scalability:

- The AWS Event Bridge has throttle limit of 2,250 transactions per second and that only counts the events after filtered by the rules. So only create and update release events are counted towards the limits. It is very unlikely that the number will be exceeded.
- Similarly, a Lambda Function can process many requests per second and is unlikely to be overwhelmed. However, if we need to scale it up to match the AWS Event Bridge we can add more servers behind a load balancer.