Table of Contents

[Introduction 2](#_Toc151147262)

[Purpose 2](#_Toc151147263)

[Requirements 2](#_Toc151147264)

[Architectural Representation 3](#_Toc151147265)

[AWS Services required for Presentation Manager and RTE 3](#_Toc151147266)

[Tech Stack 3](#_Toc151147267)

[Solution for Presentation Manager 3](#_Toc151147268)

[Solution for RTE 4](#_Toc151147269)

[Deployment 4](#_Toc151147270)

[DevOps Approach 4](#_Toc151147271)

[DevOps Cloud Security for Application 5](#_Toc151147272)

[Application and Infrastructure Monitoring 6](#_Toc151147273)

# Introduction

## Purpose

This document provides a comprehensive architecture overview of the Presentation Manager and Viatris Email Agent. It is intended to capture and convey the different architectural decisions which has been made.

## Requirements

There are two components which are integrated with MyForce app, and modules that Sales Reps use during their interaction with HCPs.

**Presentation Manager**

Development of a simple application that allows admin users to upload a file in AWS S3 for a presentation record and save the S3 link into the Salesforce, to be hosted in Viatris AWS.

|  |  |
| --- | --- |
| **Req. ID** | **Web App Requirement** |
| 1 | User with admin access shall be able to login with salesforce username and password. Integration with Salesforce for the login flow. |
| 2 | Show the presentation records in Presentation Manager user interface using an existing Salesforce Rest API |
| 3 | Allow the admin users to upload the file (Supported file type: zip and max file size :100 GB) for a presentation record. In case a file is already uploaded for a record then an attempt to upload a new file will override the existing file. |
| 4 | Save the uploaded file in AWS S3 bucket. Allowed upto a max of 100 MB. |
| 5 | Generate a signed URL for the saved zip file |
| 6 | Update the salesforce presentation record with the signed URL using an existing Salesforce REST API. |

**Viatris Email Agent (RTE)**

Development of multi-channel content experience platform for sending email and save email event information for Business accounts/Organization contacts.

|  |  |  |
| --- | --- | --- |
| Req. ID | App requirements | User Interface/ backend |
| 1 | Users shall be able to login with salesforce username and password. Integration with Salesforce for the login flow. | User Interface |
| 2 | Maintain the mapping of med rep email address to certain other Viatris domain. | User Interface |
| 3 | MyForce would use RTE REST API to send the email content (call triggered from MyForce to RTE) | Backend |
| 4 | MyForce would use RTE REST API to fetch the email event data (call triggered from MyForce to RTE) and save it to the Salesforce. | Backend |

# Architectural Representation

## AWS Services required for Presentation Manager and RTE

A screenshot of a computer

Description automatically generated

## Tech Stack

## Solution for Presentation Manager

The Presentation Manager application would a simple monolith .Net Application. The application would use SalesForce SSO feature for authentication and authorization.

The entire application would be deployed to a EC2 instance. Route 53 would be used for Callback URL for SSO enabled login. Salesforce does not accept IP, but a valid public domain as its accepted callback url.

AWS S3 would be used as a Storage Manager for all the files that are uploaded from this application. AWS Client SDK would be used in the .Net application to facilitate upload of files in to S3 bucket.

All the information relevant to the Presentation Manager Application would be saved in SQL Server. SQL Server would serve as the RDBMS for both the Presentation Manager and RTE Application.

## Solution for RTE

RTE application would be microservices platform in .NET framework, RDBMS being SQL Server.

AWS API gateway would be used to maintain all the API Endpoints for the REST APIs.

The Rest API would trigger SES (Simple Email Service) to send emails.

The SNS (Simple Notification Service) would trigger every time the email is sent using the SES services. It would track ***sends, failure, rejects, delivered, bounce, spam, delivery delays, subscriptions, open, click***.

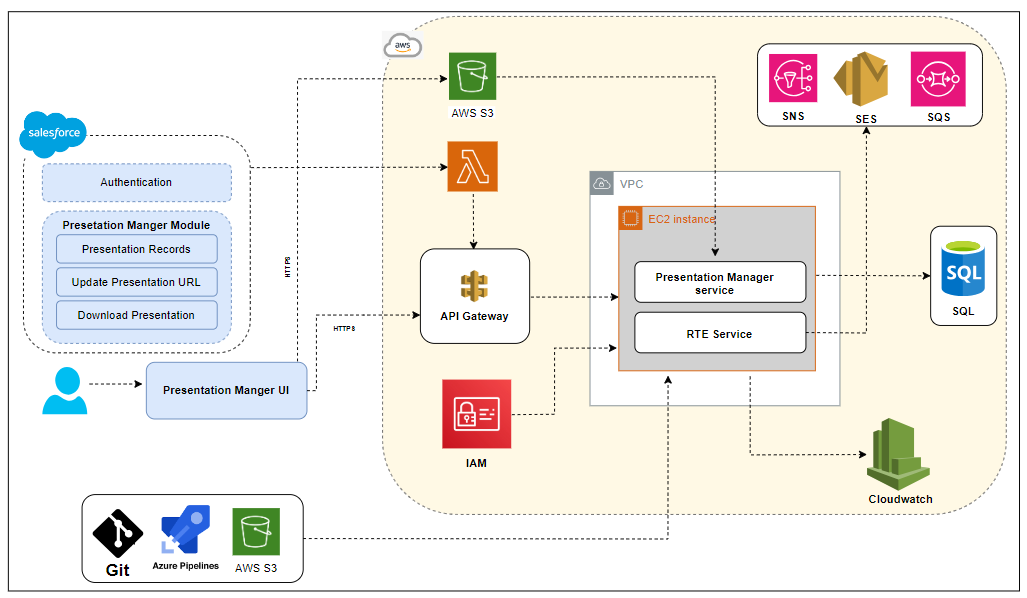
The SQS (Simple Queue Service) would push the email events and the event processor will process and save in the DB (SQL Server)

The Authorizer Lambda will help authorize the JWT token from the Salesforce login.

# Deployment

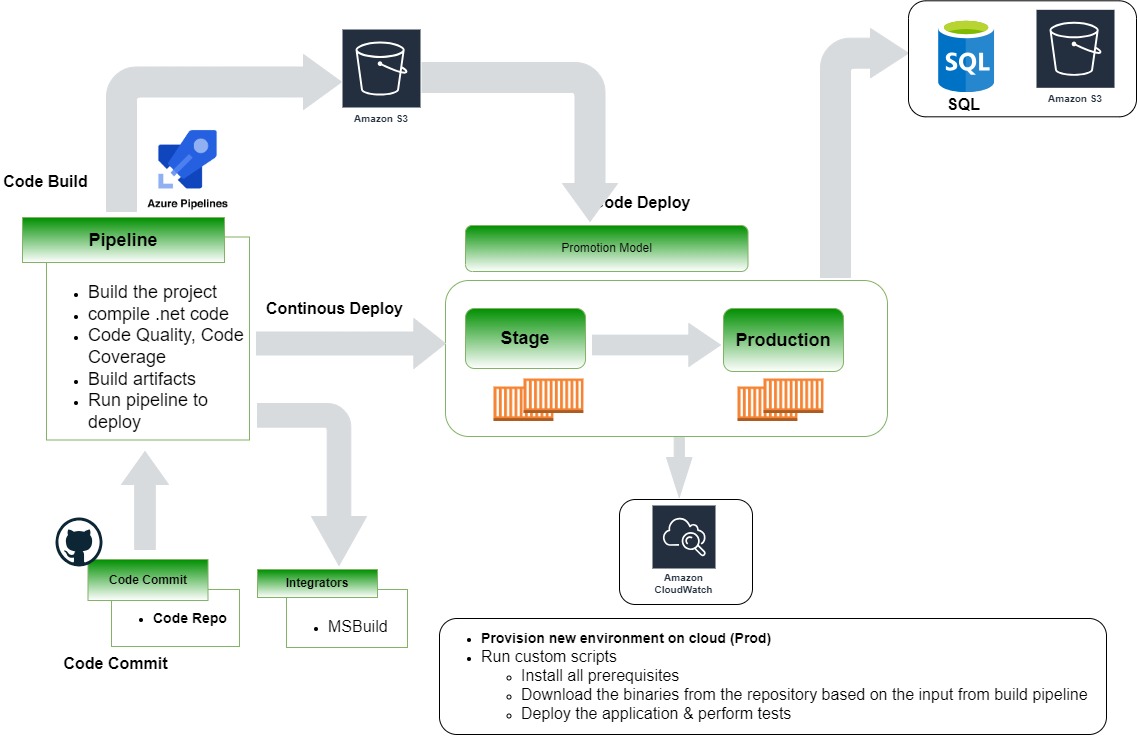
# DevOps Approach

The architecture is being approached with the following AWS services integrated across the platform. The Continuous Integration and Continuous Deployment is configured either with the proposed tools or with the existing tools.



* Set up Azure DevOps pipeline on a server to manage the deployments.
* Creating the build pipeline for the .NET application.
* Configure the build pipeline to build the application code and generate artifacts.
* Setting up the release pipeline in Azure DevOps.
* Add the build pipeline as the artifact source for the release pipeline.
* Configuring deployment stages for Staging & Production.
* Add deployment tasks to the release pipeline for deploying the .NET application.
* Release the pipeline to deploy the .NET application.
* Implement IAM roles and policies to manage access and permissions for different users and services and assign appropriate IAM roles to individuals based on their responsibilities and access requirements.
* Configuring API Gateway to manage API endpoints. It helps create, publish, maintain, monitor, and secure REST, HTTP, and WebSocket APIs at any scale.
* The S3 bucket will be configured with the IAM roles for access for specific users.
* The CloudWatch will be configured for all monitoring purposes.
* SQL Server as used to store the presentation manager RTE service application data.
* Simple email services, simple queue services, and Simple Notification Services will be configured as part of the deployment.

DevOps CI/CD Approach



The following are the steps execution as part of the pipeline the Azure DevOps pipeline is configured with all the requirement stages for the build execution:

* Azure DevOps Pipeline is configured with a multi-stage build process for the pipeline build execution.
* Continuous Integration of the build process starts and triggers the staged build process.
* The triggered build process on the compilation is successful proceed with the next gated check with the code quality.
* During the process, the image is validated for all the security integration with the scanning using the tool.
* The rules are being set with the expectation for the code to be passed in and uploaded to the S3.
* On successful compilation the build is uploaded onto the S3.
* The artifact is then called using the Continuous Integration process for the MSBuild deployment onto the EC2 instance.

## DevOps Cloud Security for Application

* Separate VPC will be configured and deploy the microservices and application in the private subnet.
* Security group Rules for the private and public configured.
* Access Identity and Management (IAM) Access Analyzer.
* Secrets Manager that helps to securely store, manage, and retrieve secrets for your applications and services.
* Security assessments on the NACL rules & and configuration.
* Configuring the right access to the corresponding repos, S3, and other services.
* Software upgradation and other security patches installation will be taken care.

## Application and Infrastructure Monitoring

* Configure monitoring tools like AWS CloudWatch to gain insights into the performance and health of the infrastructure and applications.
* Set up monitoring alerts, dashboards, and anomaly detection to proactively identify and resolve issues.
* Monitor key metrics such as CPU usage, memory utilization, response times, and error rates.