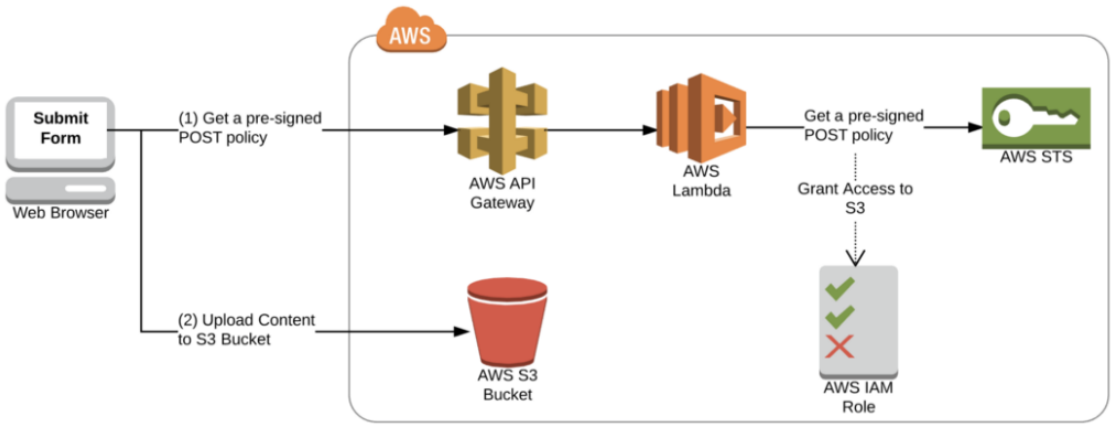
Upload Image



Serverless application architecture for the above use case which involves direct upload of image to AWS S3 bucket from the browser

Benefits:

- Improved performance
- Fault tolerant
- Less overhead over back-end servers to scale

Flow:

User clicks on “Choose File” and then on “Upload image”

- Browser hits AWS API gateway, which then triggers AWS lambda funciton to get pre-signed POST URL for upload on S3 bucket making use of specific IAM roles
- Browser then makes a POST request to S3 directly on the URL received in previous step, to upload the image

API(Deployed on AWS API Gateway):

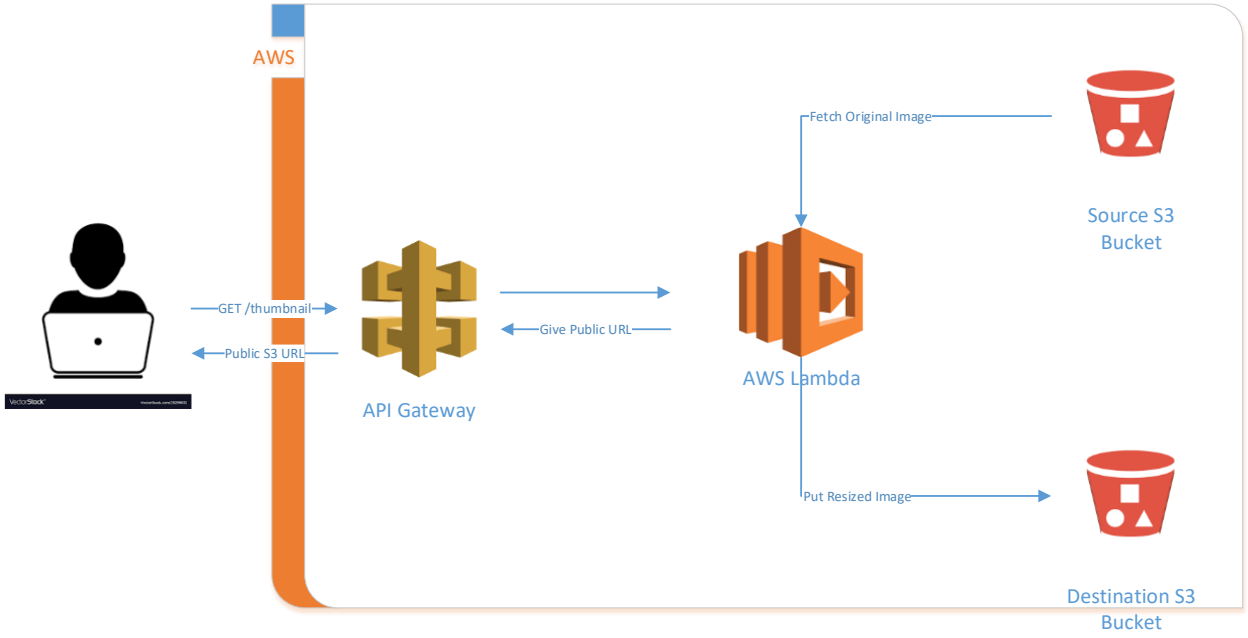
GET: /uploadURL

Description: Returns pre-signed POST URL for S3

Setup:

- AWS Lambda to have S3 access policy
- CORS policy on S3 bucket
- S3 bucket to be public
- Lambda runtime in NodeJS

Generate Thumbnail



Serverless application architecture for the above use case

Flow:

After succesful image upload, user selcets the desired resolution from the dropdown and clicks on “Generate Thumbnail”:

- Browser hits AWS API gateway, which then triggers AWS lambda funciton
- Lambda function retrieves image from the source S3 bucket
- Resizes image using Sharp(javascript image processing library)
- Stores the thumbnail in destination bucket and makes it public
- Returns the publicly accessible URL

API(Deployed on AWS API Gateway):

GET: /thumbnail

QueryParams:

imageName: Name of iamge in source S3 Bucket

String

Required

res: Desired resolution size

String

Required

Description: Returns publicly accesible thumbnail URL

Setup:

- AWS Lambda to have S3 access policy
- CORS policy on S3 buckets
- S3 buckets to be public
- Lambda runtime in NodeJS