CloudTopia Passport Office – Problem Introduction

GOALS

1. Automate Manual Effort
2. Notifications
3. API

CloudTopia Passport Office

Cloudtopia Citizens

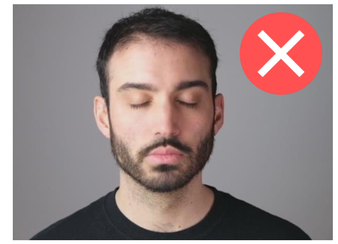
Upload

Web Server

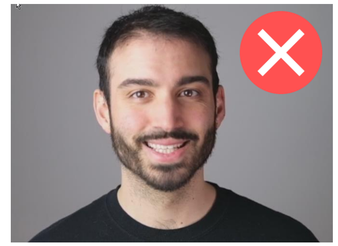
Evaluation

Approve

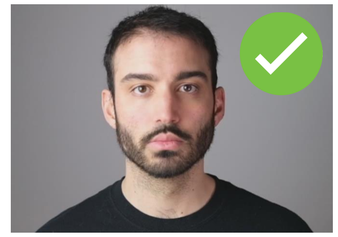
Reject



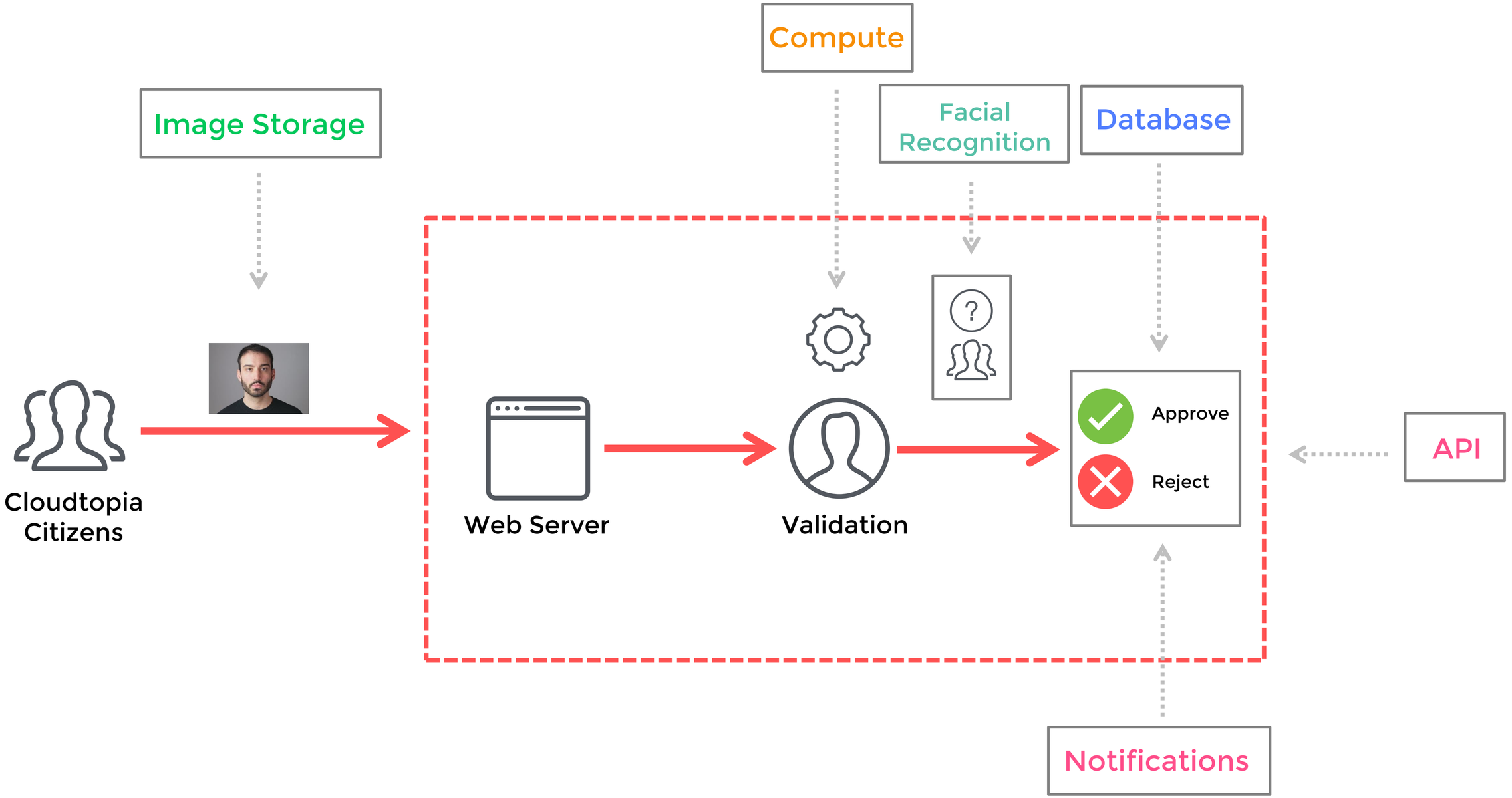
Eyes Closed



Smiling



CloudTopia Passport Office – Components



Core Technology Choices

Image Storage

Needs: Blob file storage

Options:

Amazon Elastic File System (Amazon EFS)

Amazon Simple Storage Service (Amazon S3)

Compute

Needs: Event Processing, API

Options:

Amazon Elastic Compute Cloud (Amazon EC2)

AWS Fargate

AWS Lambda

Facial Recognition

Needs: Face Detection & Analysis

Options:

Amazon Rekognition

Database

Needs: Key/Value Record Storage

Options:

Amazon Relational Database Service (Amazon RDS)

Amazon Aurora

Amazon DynamoDB

Notifications

Needs: Message Publishing

Options:

Amazon Simple Queue Service (Amazon SQS)

Amazon EventBridge

Amazon Simple Notification Service (Amazon SNS)

API

Needs: Data Retrieval

Options:

AWS AppSync

Amazon API Gateway

Core Technology Choices

Image Storage

Amazon Simple Storage Service (Amazon S3)

Compute

AWS Lambda

Facial Recognition

Amazon Rekognition

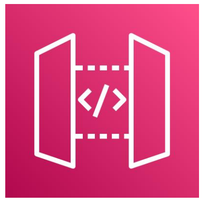
Database

Amazon DynamoDB

Notifications

Amazon Simple Notification Service (Amazon SNS)

API

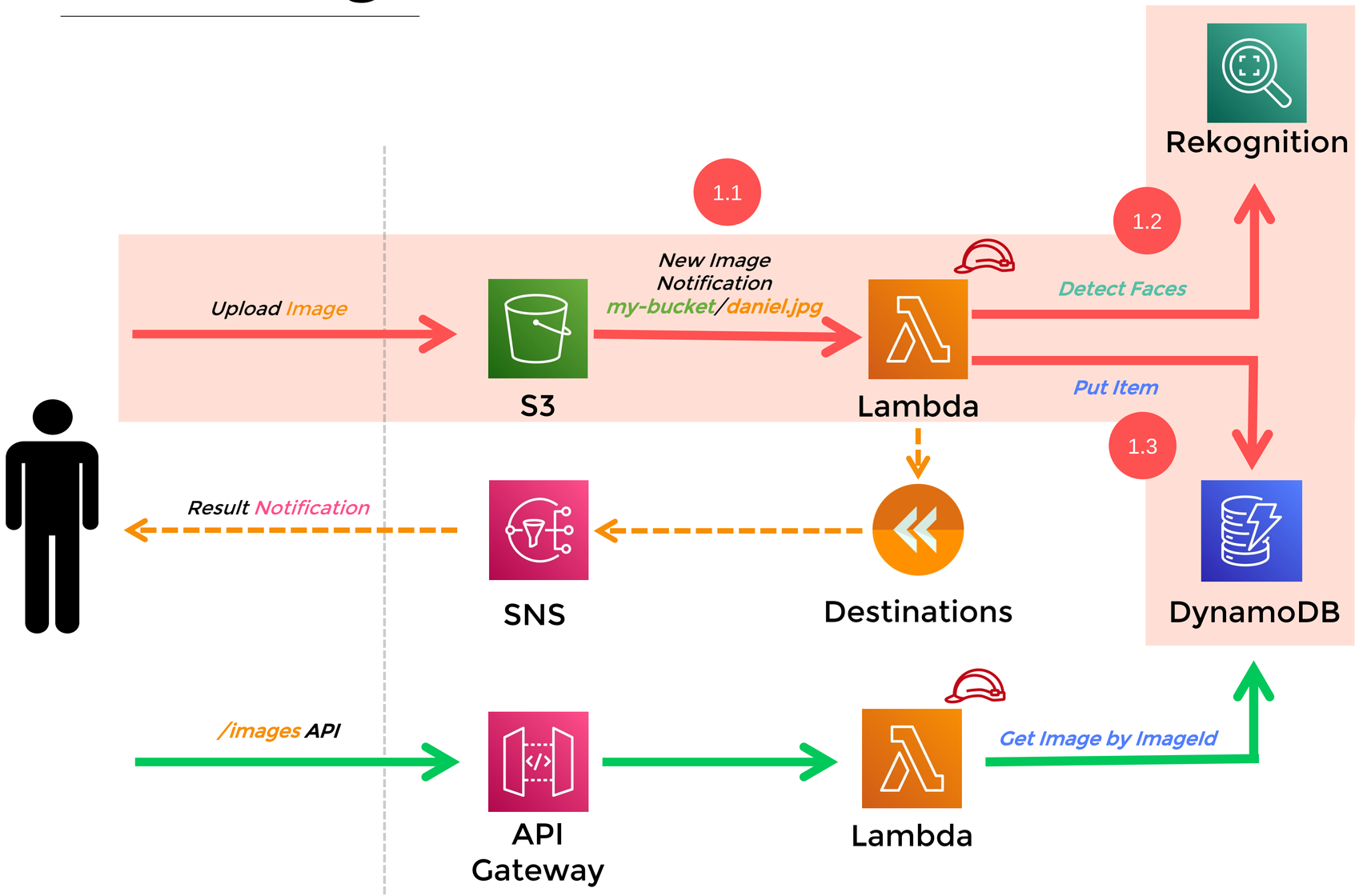


Amazon API Gateway

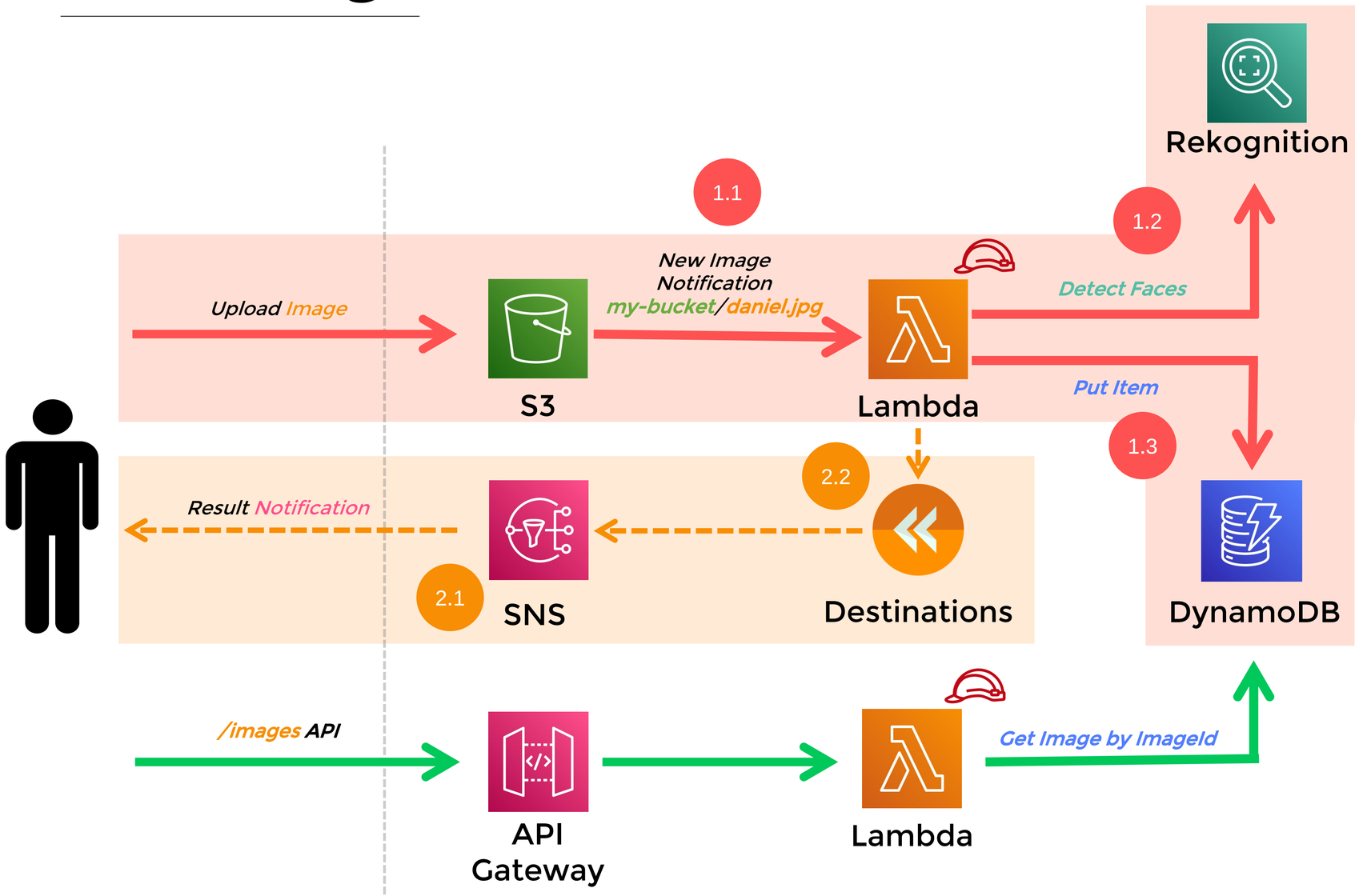
End Design

Part 1 Data Processing Flow

- 1.1 S3 + Lambda
- 1.2 Rekognition
- 1.3 DynamoDB



End Design



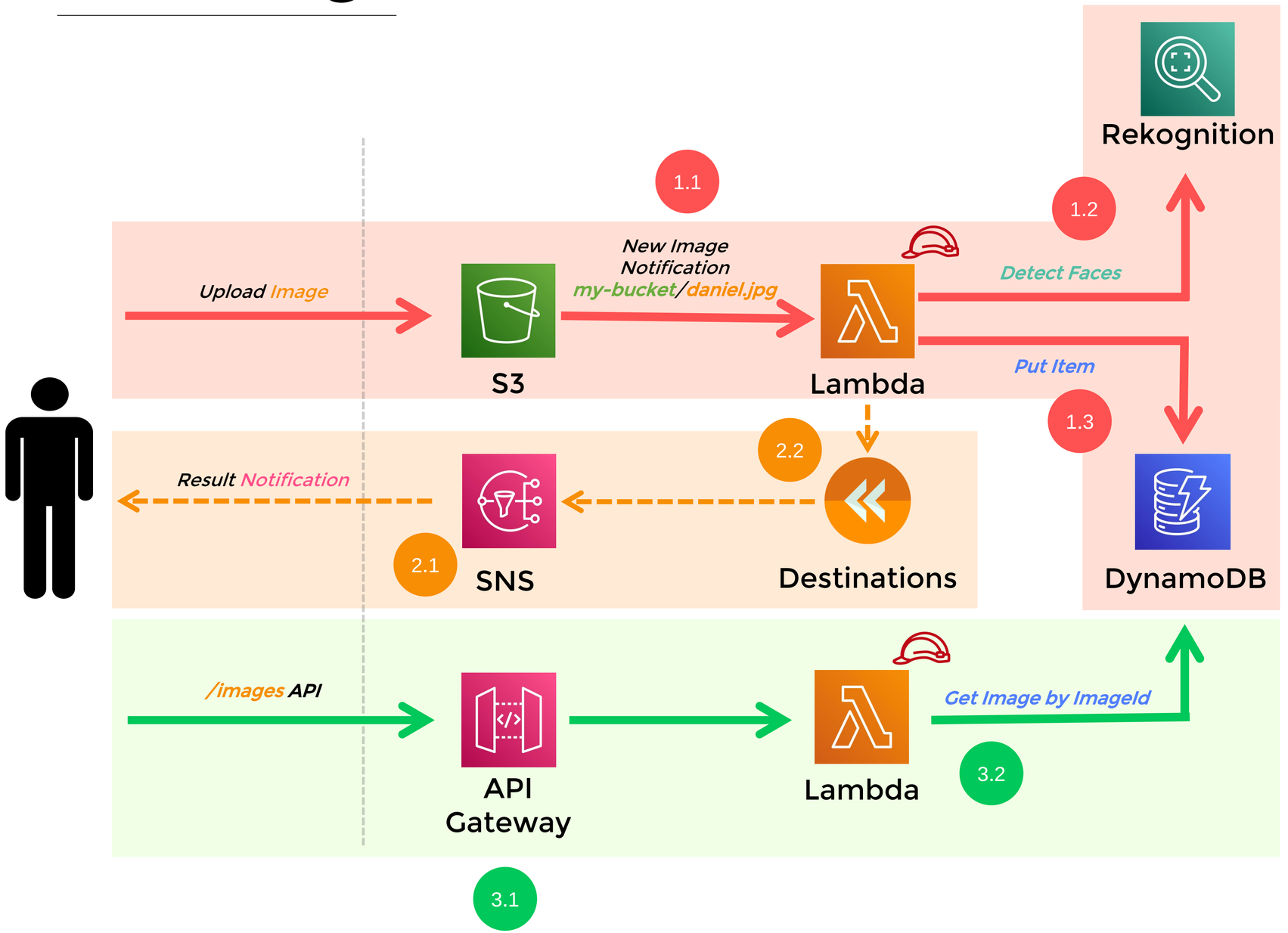
Part 1 Data Processing Flow

- 1.1 S3 + Lambda
- 1.2 Rekognition
- 1.3 DynamoDB

Part 2 Notifications Flow

- 2.1 SNS
- 2.2 Destinations

End Design



Part 1 Data Processing Flow

- 1.1 S3 + Lambda
- 1.2 Rekognition
- 1.3 DynamoDB

Part 2 Notifications Flow

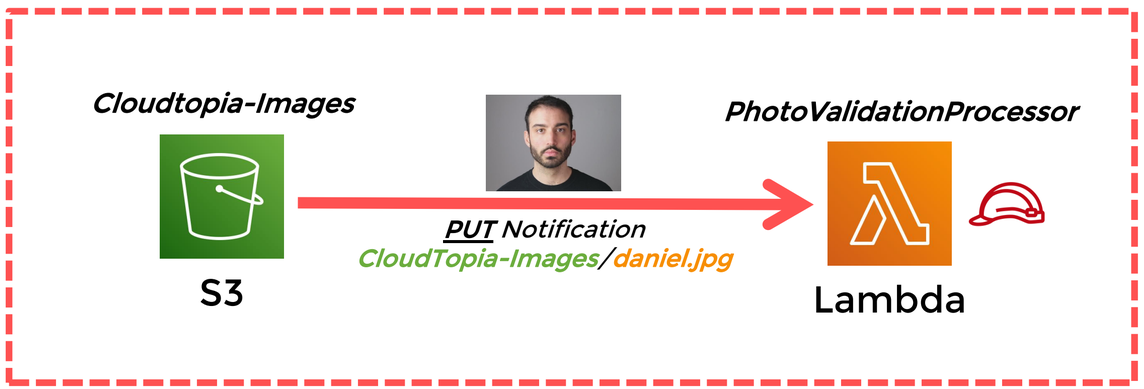
- 2.1 SNS
- 2.2 Destinations

Part 3 Data Retrieval Flow

- 3.1 API Gateway
- 3.2 Lambda

Cloudtopia Part 1.1 – S3 + Lambda

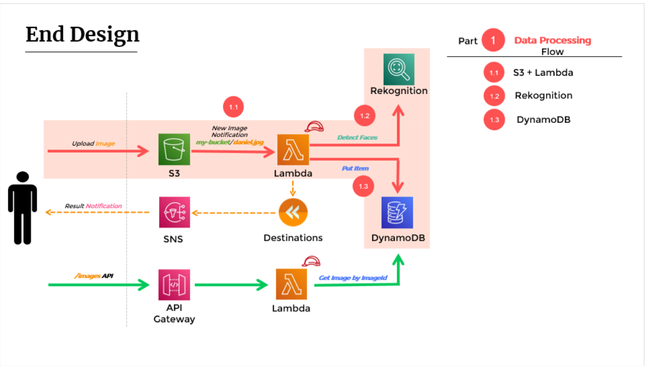
```
def lambda_handler(event, context):
```



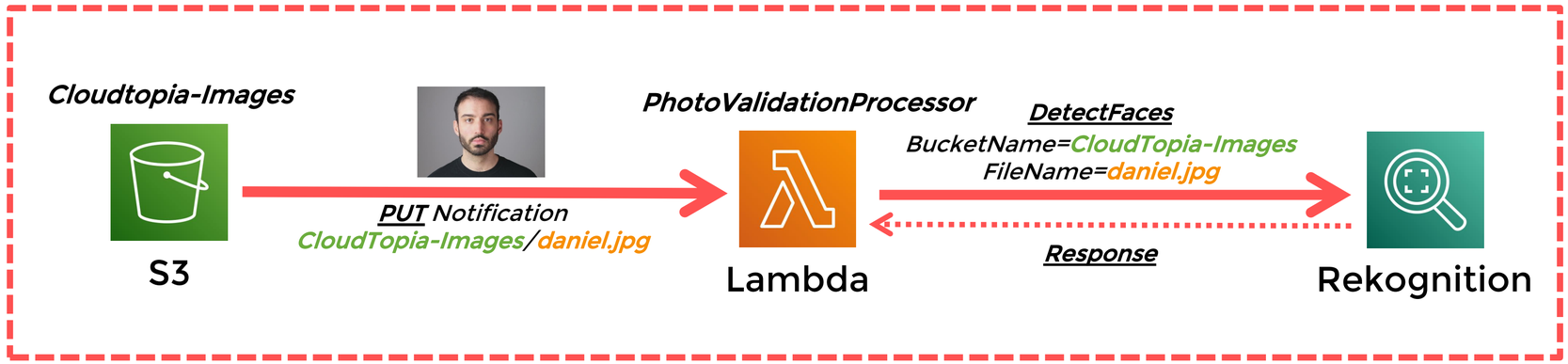

```
{
   "Records":[
      {
         "s3":{
            "bucket":{
               "name":"CloudTopia-Images",
               "arn":"arn:aws:s3:::CloudTopia-Images"
            },
            "object":{
               "key":"daniel.png",
               "size":441027
            }
         }
      }
   ]
}
```

Steps

1. Create S3 Bucket CloudTopia-Images
2. Create Lambda PhotoValidationProcessor
3. Create PUT Trigger between CloudTopia-Images and PhotoValidationProcessor
4. Attach IAM Policy *S3:GetObject* to PhotoValidationProcessor



Cloudtopia Part 1.2 – Rekognition

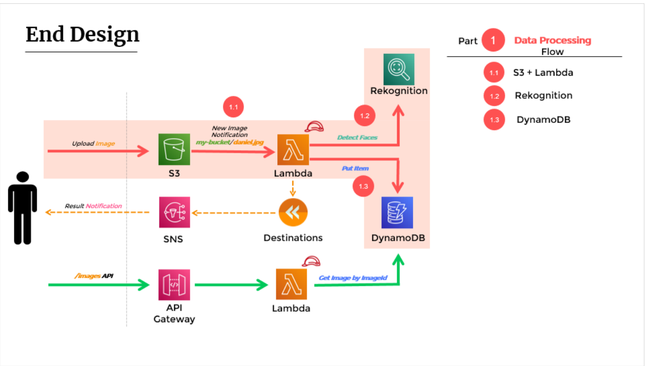


Response

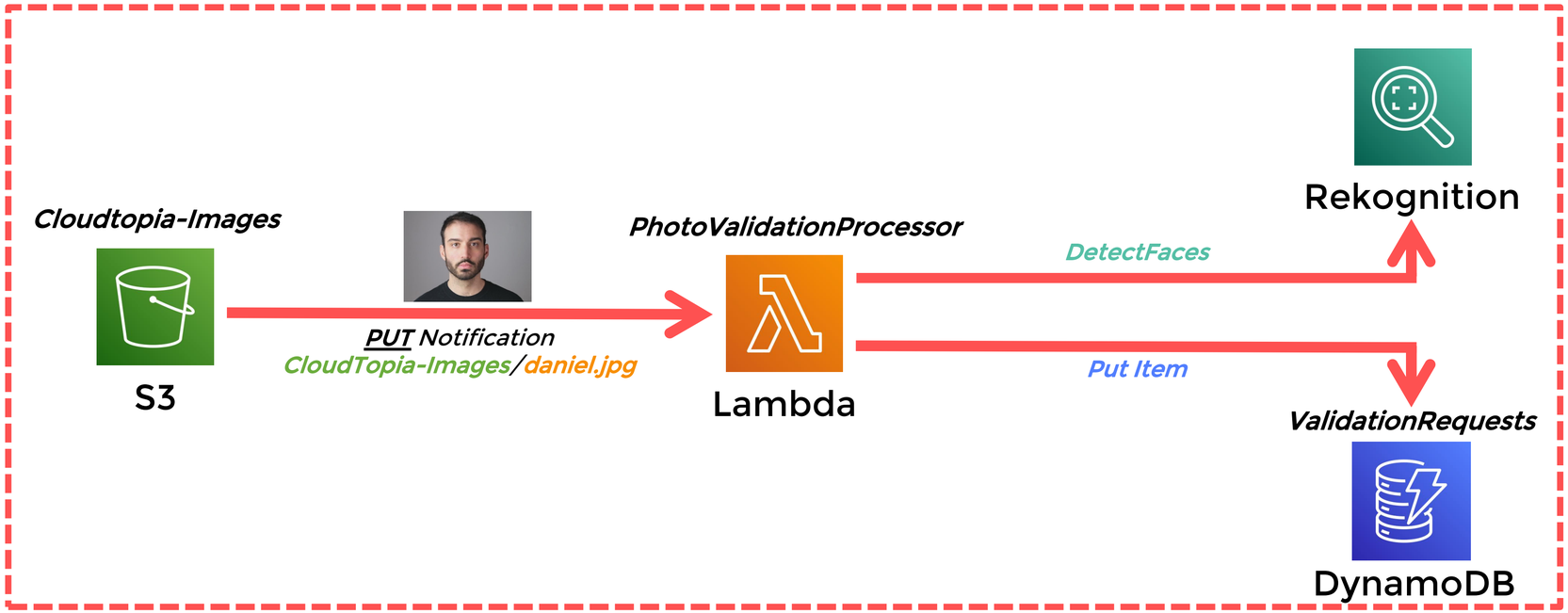

```
"FaceDetails": [
    {
        "Smile": {
            "Value": false,
            "Confidence": 94.74140930175781
        },
        "Sunglasses": {
            "Value": false,
            "Confidence": 99.99543762207031
        },
        "EyesOpen": {
            "Value": true,
            "Confidence": 96.44212341308594
        },
        "MouthOpen": {
            "Value": false,
            "Confidence": 92.9201431274414
        },
        "Emotions": [
            {
                "Type": "CALM",
                "Confidence": 99.008056640625
            }
        ]
    }
]
```


Steps

1. Attach IAM Policy *Rekognition:DetectFaces* to PhotoValidationProcessor
2. Extract file name from S3 PUT Event
3. Call Rekognition ***DetectFaces*** API
4. Extract Value/Confidence for Smile, Sunglasses, EyesOpen, MouthsOpen, Calm Emotion
5. Apply Pass/Fail **Business Logic** (Confidence >= 90.00 && Value)

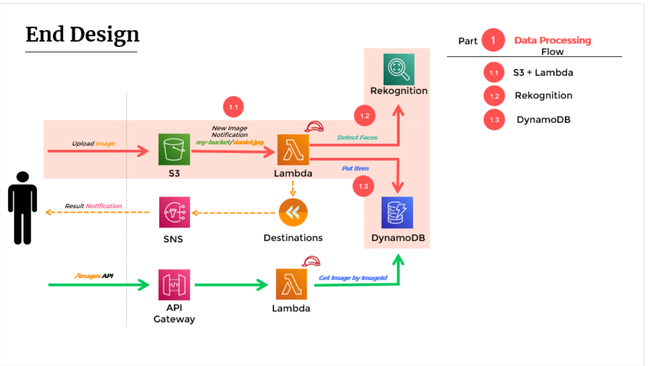


Cloudtopia Part 1.3 – DynamoDB

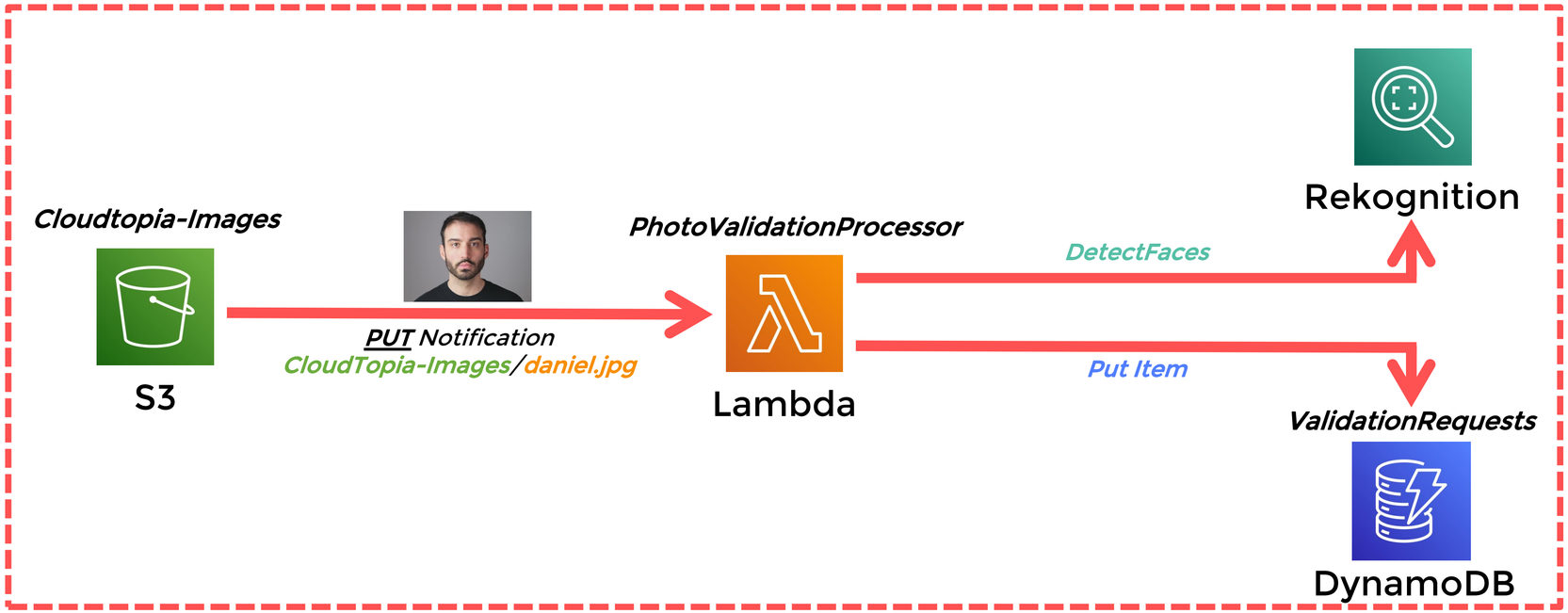


Steps

1. Decide Data Model & DynamoDB Table Schema
2. Create DynamoDB ValidationRequests Table
3. Call ***PutItem*** API
4. Attach IAM Policy *DynamoDB:PutItem* to PhotoValidationProcessor

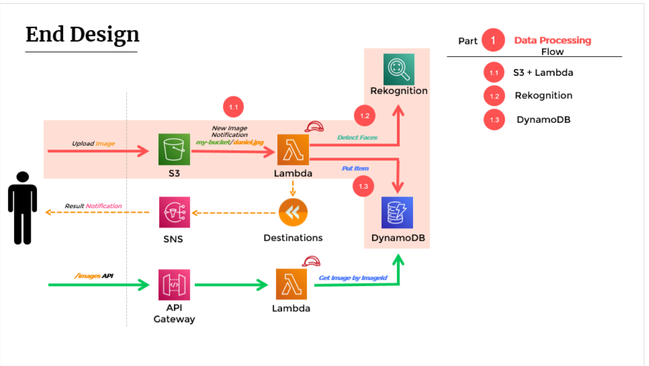


Cloudtopia Part 1.3 – DynamoDB



Steps

1. Create DynamoDB ValidationRequests Table
2. Call ***PutItem*** API
3. Attach IAM Policy *DynamoDB:PutItem* to PhotoValidationProcessor



Cloudtopia Part 1.3 – DynamoDB

1 Decide Data Model & DynamoDB Table Schema

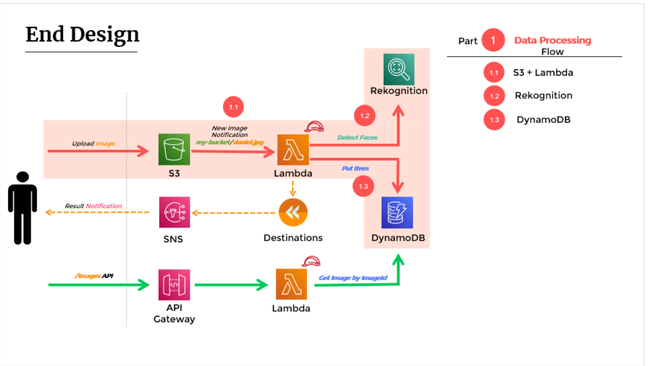
ValidationRequests

			Example
PK	FileName :	String	daniel.jpg
	ValidationResult :	String	PASS or FAIL
	FailureReasons :	List<String>	[Smile, EyesOpen]
	Timestamp :	String	2023-01-29T01:20:45
	FileLocation :	String	CloudTopia-Images/daniel.jpg
	FaceDetails :	String (JSON)	{ “Smile”: { “Value”: false, “Confidence”: 99.957848378}, … }

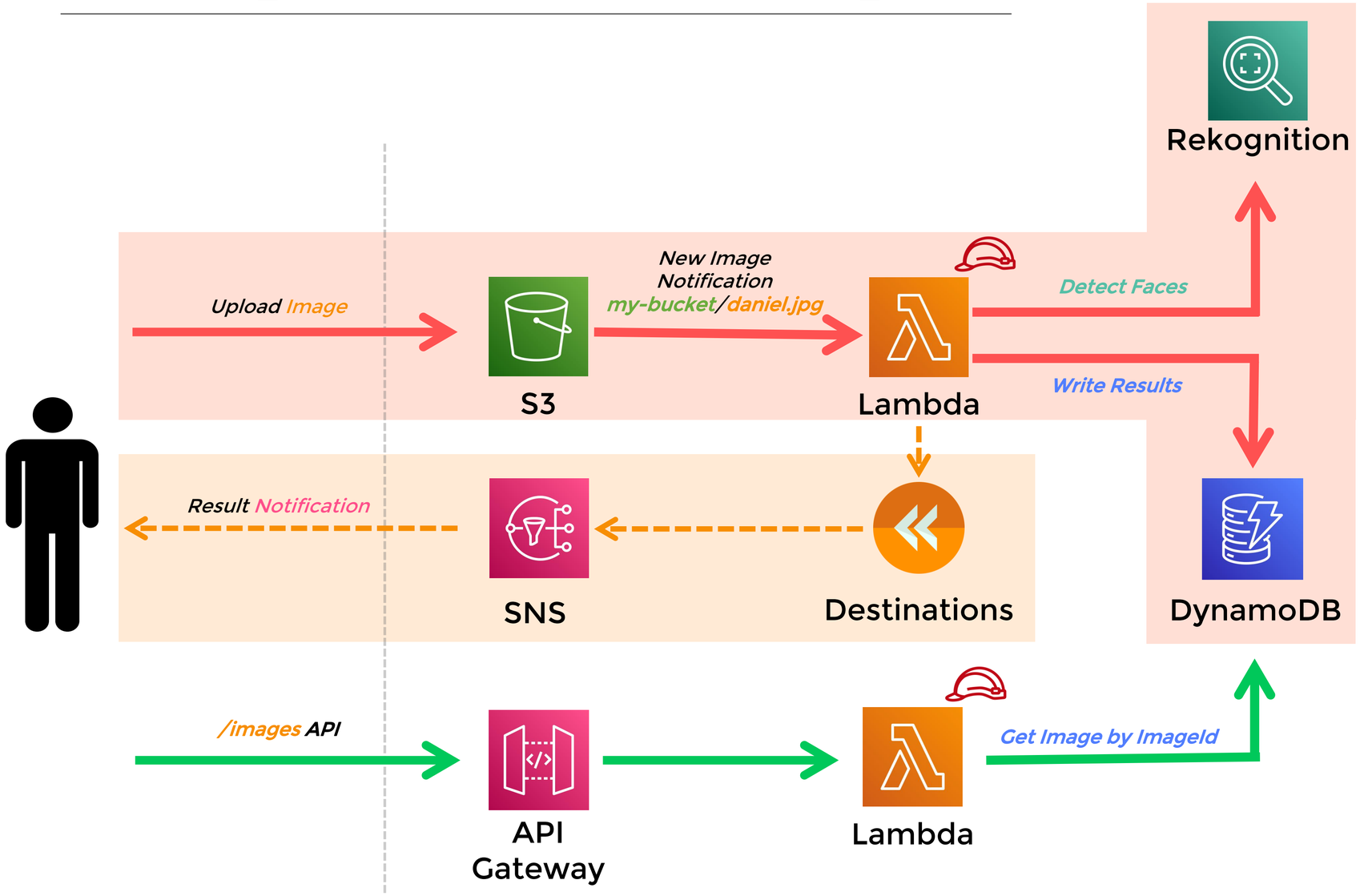
2 Create DynamoDB ValidationRequests Table

3 Call ***PutItem*** API

4 Attach IAM Policy *DynamoDB:PutItem* to PhotoValidationProcessor



Checkpoint – Part 1 Complete



Part 1 Data Processing Flow

- 1.1 S3 + Lambda
- 1.2 Rekognition
- 1.3 DynamoDB

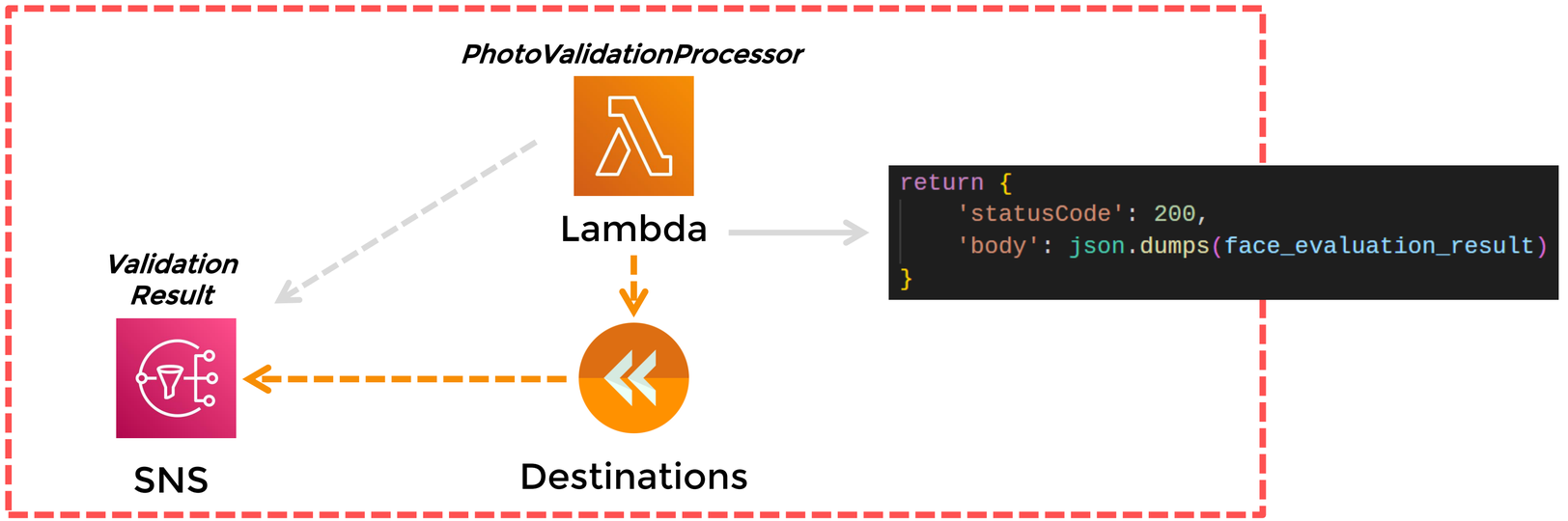
Part 2 Notifications Flow

- 2.1 SNS
- 2.2 Destinations

Part 3 Data Retrieval Flow

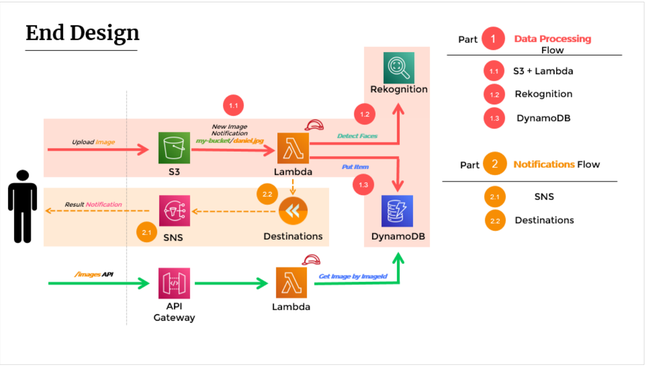
- 3.1 API Gateway
- 3.2 Lambda

Cloudtopia Part 2 – SNS and Destinations

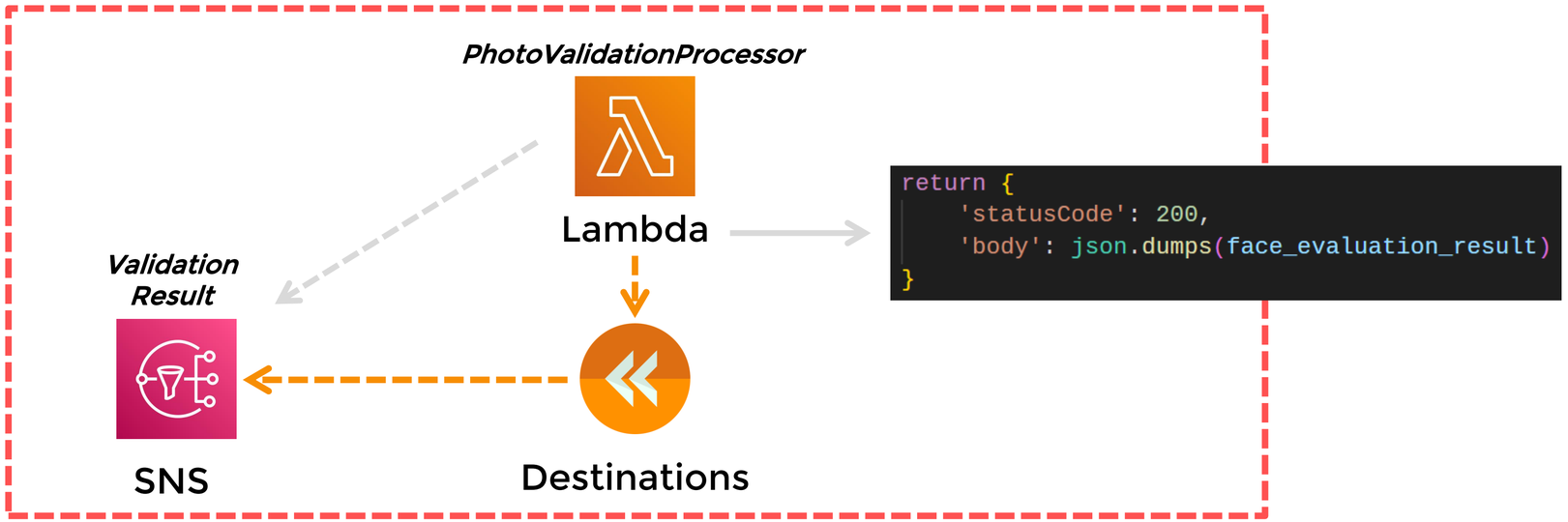


Steps

1. Create *ValidationResult* SNS Topic
2. Attach IAM Policy *SNS:Publish* to PhotoValidationProcessor
3. Modify PhotoValidationProcessor to return valid HTTP Response
4. Configure Lambda Destinations to deliver result to ValidationResult SNS Topic

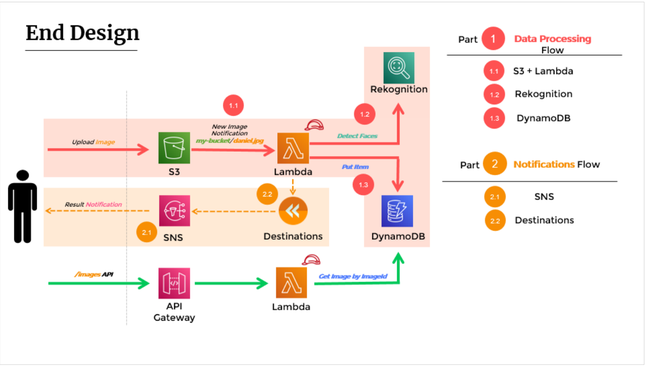


Cloudtopia Part 2 – SNS and Destinations

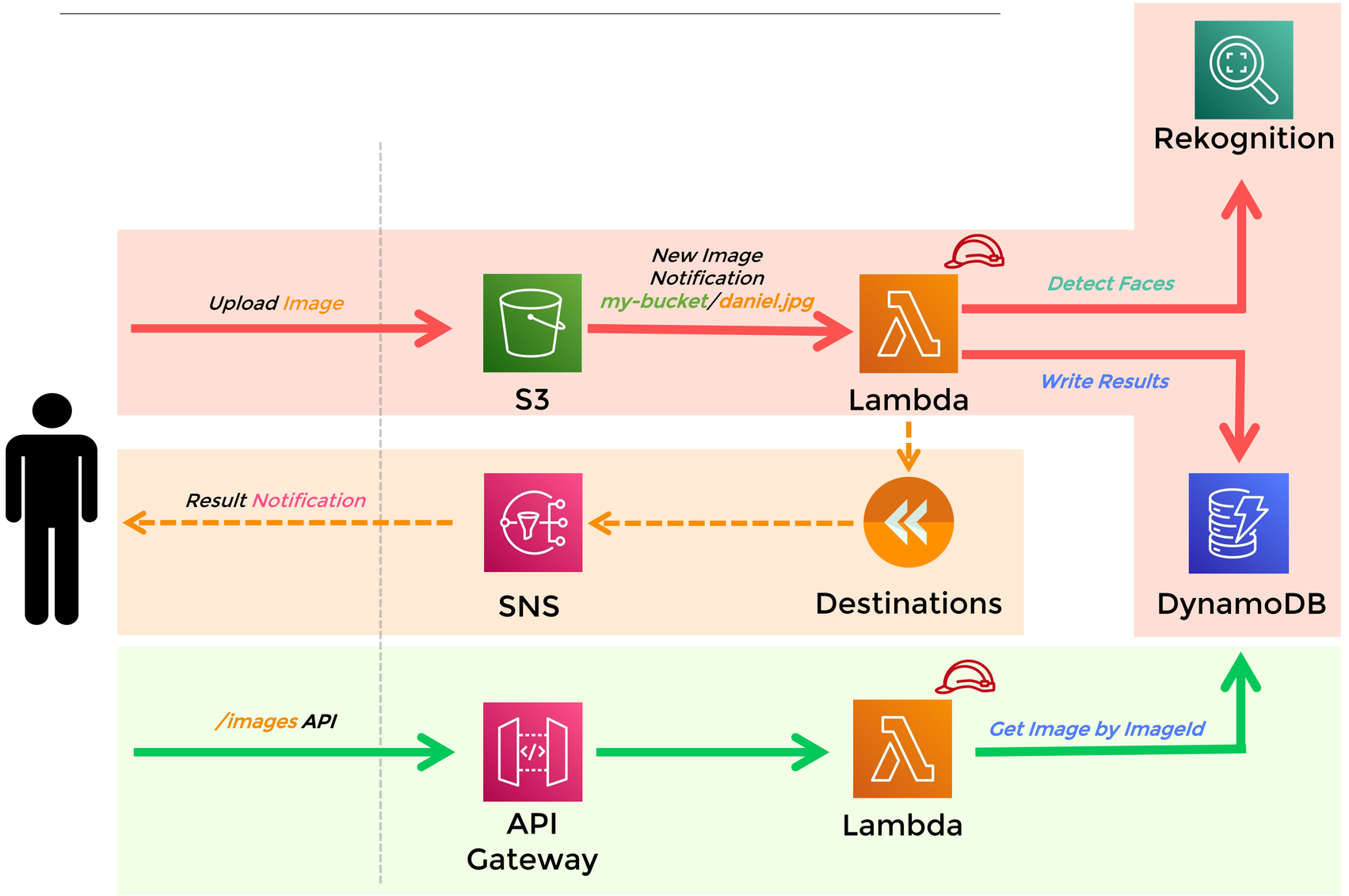


Steps

1. Attach IAM Policy *SNS:Publish* to PhotoValidationProcessor
2. Modify PhotoValidationProcessor to return valid HTTP Response
3. Configure Lambda Destinations to deliver result to ValidationResult SNS Topic
4. Configure Lambda Destinations to deliver result to ValidationResult SNS Topic



Checkpoint – Part 2 Complete



Part 1 Data Processing Flow

- 1.1 S3 + Lambda
- 1.2 Rekognition
- 1.3 DynamoDB

Part 2 Notifications Flow

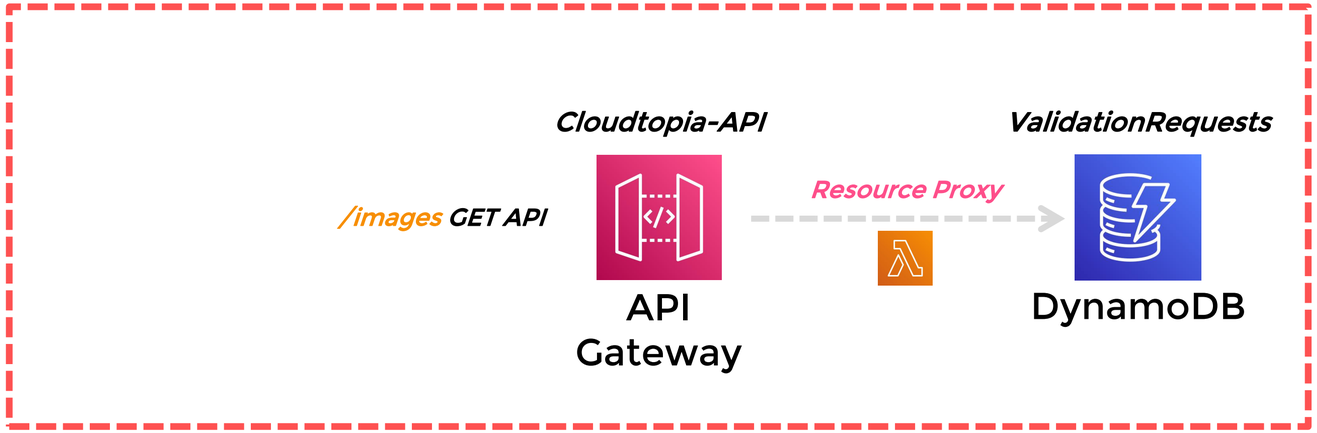
- 2.1 SNS
- 2.2 Destinations

Part 3 Data Retrieval Flow

- 3.1 API Gateway
- 3.2 Lambda

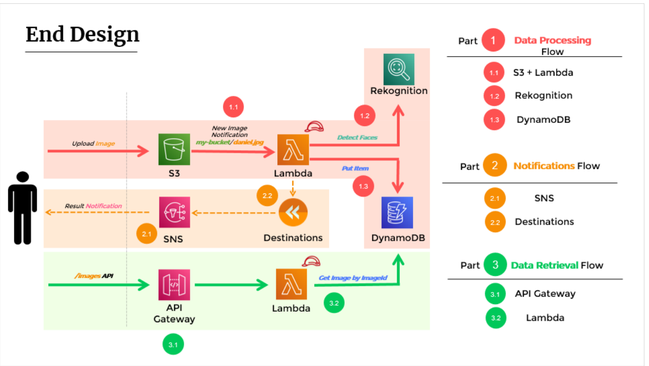
Cloudtopia Part 3.1 – API Gateway

HTTP API

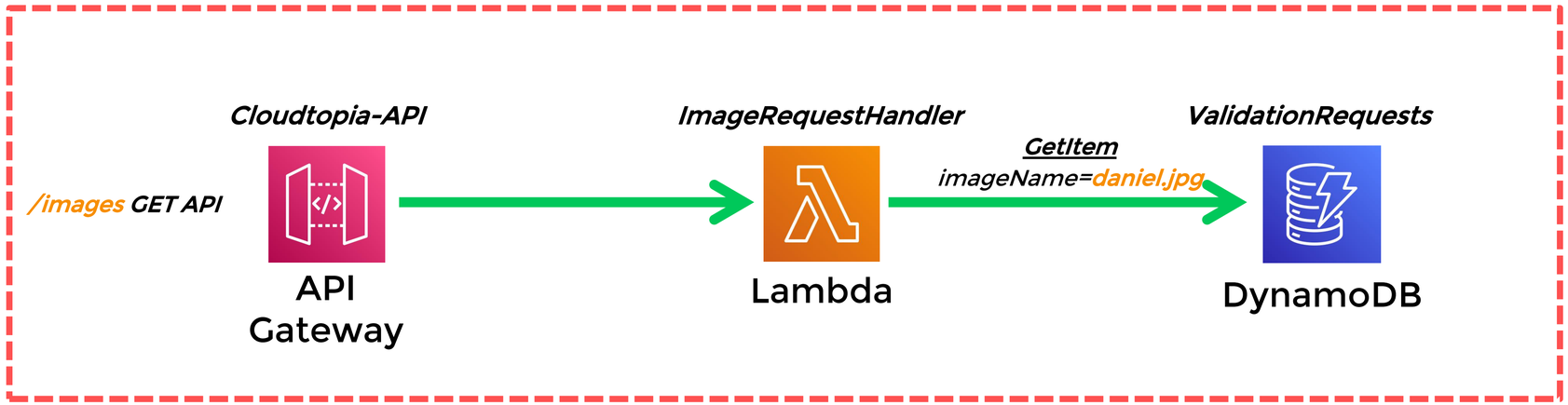


Steps

1. Create HTTP API
2. Create /images GET Route

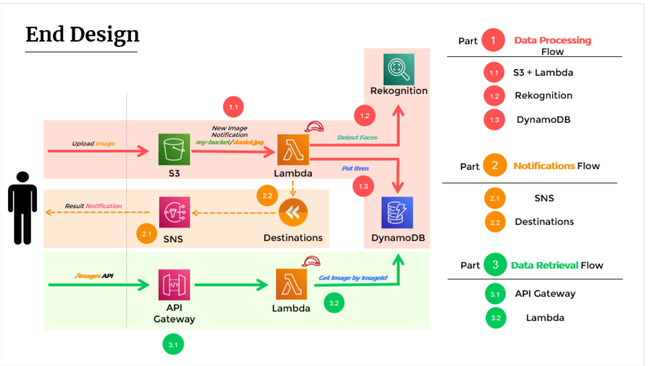


Cloudtopia Part 3.2 – Lambda

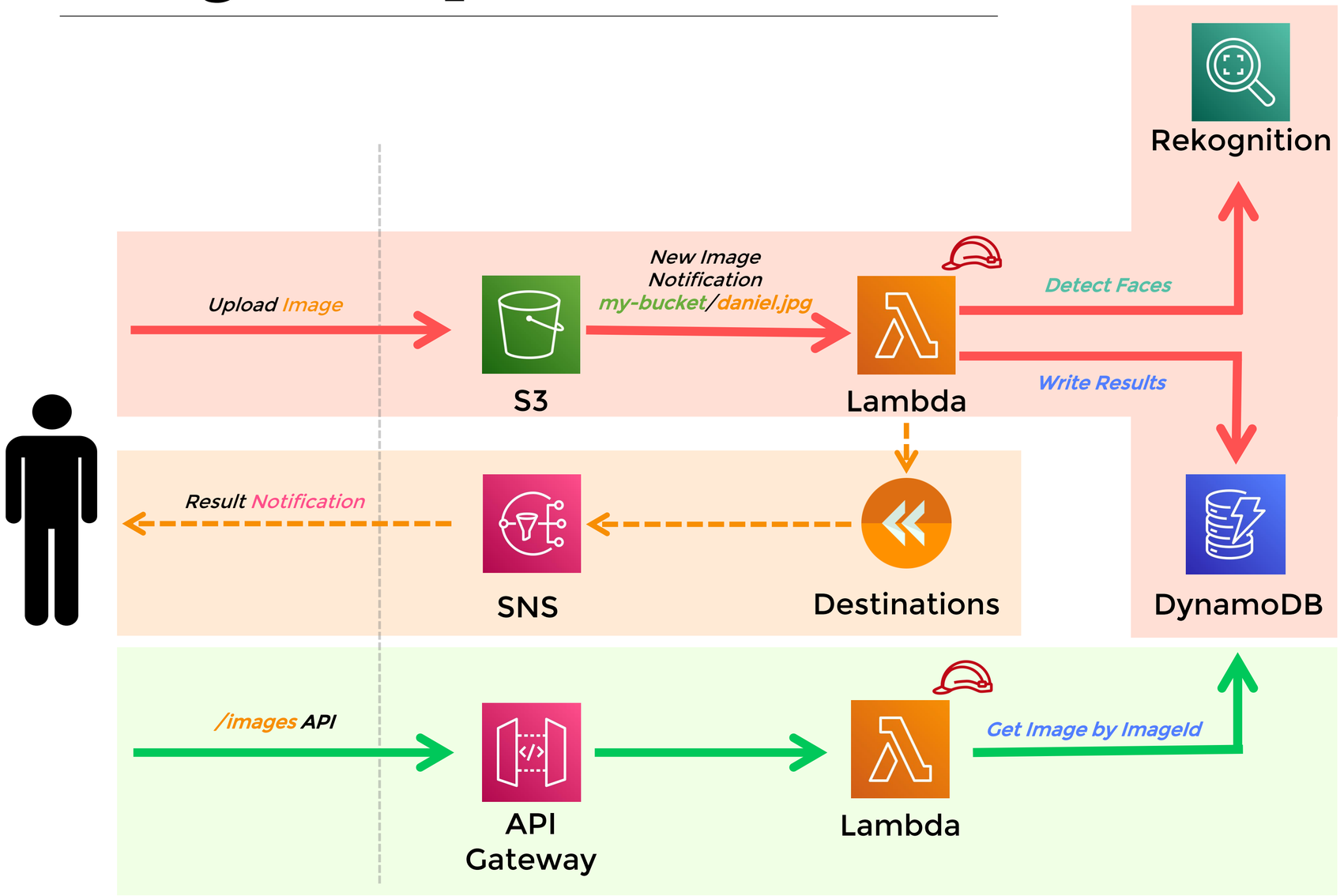


Steps

1. Create new Lambda Function ImageRequestHandler
2. Associate API with ImageRequestHandler
3. Attach IAM Policy *DynamoDB:GetItem* to ImageRequestHandler
4. Modify code to fetch ValidationRequest and return result



Design Complete



Part 1 Data Processing Flow

- 1.1 S3 + Lambda
- 1.2 Rekognition
- 1.3 DynamoDB

Part 2 Notifications Flow

- 2.1 SNS
- 2.2 Destinations

Part 3 Data Retrieval Flow

- 3.1 API Gateway
- 3.2 Lambda